МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний авіаційний університет

PROFESSIONAL ENGLISH FUNDAMENTALS OF SOFTWARE ENGINEERING

Рекомендовано
Міністерством освіти і науки України
як навчальний посібник
для студентів вищих навчальних закладів

Київ 2015

УДК 811.111:004.4(075.8)
ББК Ш143.21-913
Р93

Автори: *О.М. Акмалдінова, О.Є. Бугайов, Л.Г. Теремінко, О.О. Гурська, Т.А. Мислива, Н.І. Муркіна*

Рецензенти: *О.С. Снитко* – д-р філол. наук, проф. (Київський національний університет імені Тараса Шевченка);
С.С. Коломієць – канд. пед. наук, доц. (Національний технічний університет України «Київський політехнічний інститут»);
Т.А. Гузік – канд. пед. наук, доц. (Київський національний економічний університет імені Вадима Гетьмана)

Рекомендовано Міністерством освіти і науки України (лист № 1/11 від 24.10.2014 р.).

Professional English. Fundamentals of Software Engineering :
Р93 навч. посібник / О.М. Акмалдінова, О.Є. Бугайов, Л.Г. Теремінко та ін. – К. : НАУ, 2015. – 300 с.

ISBN 978-966-598-950-9

Посібник містить оригінальні тексти фахового змісту, які супроводжуються термінологічним тематичним вокабуляром та вправами різного методичного спрямування.

Для студентів вищих навчальних закладів.

УДК 811.111:004.4(075.8)
ББК Ш143.21-913

ISBN 978-966-598-950-9

ПЕРЕДМОВА

Навчальний посібник розроблено з метою використання на практичних заняттях з англійської мови професійного спрямування студентами комп'ютерних спеціальностей, зокрема, інженерії програмного забезпечення, а також для самостійної роботи студентів. Впродовж останніх років досягнений значний прогрес в питаннях розроблення та вдосконалення комп'ютерного програмного забезпечення, частка якого в сучасних комп'ютерних системах неухильно зростає. На сьогодні про інженерію програмного забезпечення можна говорити як про окрему вагому підгалузь комп'ютерної науки, яка сама складається з низки більш спеціалізованих субгалузей, таких як: конструювання програмного забезпечення, моделювання програмного забезпечення, архітектура програмного забезпечення, проектування програмного забезпечення та інших, що знайшло своє відображення в цьому посібнику.

До складу посібника входять дванадцять розділів, присвячених конкретним питанням інженерії програмного забезпечення. Основу кожного розділу складають автентичні тексти фахової тематики, які супроводжуються англо-українськими словниками для введення професійної лексики та комплексом передтекстових і післятекстових вправ різного спрямування (лексичних, лексико-граматичних, комунікативних), націлених на закріплення та контроль засвоєння студентами навчального матеріалу, що сприятиме їх майбутній професійній, інженерній і науковій діяльності в умовах сучасного глобалізованого світу.

Посібник завершується розділом "Supplementary Reading", який містить додаткові оригінальні тексти з поясненнями важких для розуміння та перекладу словосполучень, призначень для самостійного опрацювання під час позааудиторної роботи.

UNIT 1. SOFTWARE CONSTRUCTION

Exercise 1. *Study the basic vocabulary.*

a) terms

software – програмне забезпечення (ПЗ)
software construction – конструювання ПЗ
software engineering – інженерія ПЗ
software design – проектування ПЗ
software life cycle – життєвий цикл ПЗ
software system – система ПЗ, програмний комплекс
software project – проект ПЗ
hardware – апаратне, технічне забезпечення або оснащення *(на відміну від програмного)*; елементи комп'ютерів; *сл.* «залізо»
coding – програмування, кодування
verification – верифікація
verify – контролювати, перевіряти
unit testing – тестування компонентів системи
integration testing – тестування взаємодії компонентів системи
debugging – налагодження
routine – підпрограма
variable – змінна
formatting – подання *(інформації)* у форматі
named constant – іменована константа
control structure – керівна конструкція
statement – оператор (*мови програмування)*
low(high)-level design – низько-, високорівневий проект
source file – вихідний файл
software configuration – конфігурація програмних засобів
configuration item – елемент конфігурації
content – зміст, інформаційне наповнення
test case – набір тестових даних, контрольний приклад *(документально оформлений посібник, який визначає, як має / може бути протестована функція або комбінація функцій)*

b) nouns

creation – створення
combination – поєднання
activity – дія, операція, робота, захід

activities – діяльність
boundary – межа
integration – інтеграція, поєднання
groundwork – основа, база, фундамент
management – управління, керування

c) verbs
refer (to) – 1) стосуватися; 2) називати
involve – включати в себе, передбачати
link – з'єднувати, зв'язувати, сполучати
provide – забезпечувати, надавати
vary – міняти(ся), змінювати(ся), відрізнятися, варіюватися
depend (on) – залежати (від)
integrate – об'єднувати, інтегрувати
lay (*p., pp.* laid) – класти
proceed – 1) відбуватися; 2) продовжувати(ся)
determine – визначати
create – створювати
select – вибирати, добирати; комплектувати
organize – організовувати, створювати, впорядковувати
review – перевіряти; оглядати; рецензувати
polish – шліфувати; детально опрацьовувати; вдосконалювати
tune – регулювати, налагоджувати
produce – створювати, виробляти

d) adjectives
detailed – детальний, досконалий
meaningful – значущий
significant – значний, важливий, істотний
particular – конкретний, даний
specific – конкретний; специфічний; особливий

e) adverbs
separately – окремо
successfully – успішно
carefully – уважно, обережно
typically – зазвичай, звичайно
closely – 1) близько, тісно; 2) дуже
thus – так, таким чином

Exercise 2. *Choose verbs among the following words. Put the first letters of the verbs into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Vary, routine, enter, boundary, refer, particular, involve, subsystem, facilitate, separately, integrate, few, create, significant, access, specific, tell, instruct, success, operate, meaningful, navigate.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) refer (to), detailed, combination, meaningful, link, involve, vary, component, need, integrate, specific, verify, proceed, create, select, polish, carefully, tune, manage, strongly, typical;

b) meaningful, integration, strongly, output, carefully, separately, fast, few, closely.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: construct – construction – constructor – constructive – constructively

Construct, mean, combine, verify, integrate, act, strong, system, separate, success, add, select, care, close, detail, create, engine.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

The term software construction refers to; the detailed creation of working, meaningful software; a combination of coding, verification, unit testing, integration testing, and debugging; software engineering; detailed boundaries between design, construction, and testing; to depend upon the software life cycle processes; the integration of separately constructed routines; to lay the groundwork; to proceed successfully; to create the named constants; to select control structures; to organize blocks of statements; to integrate software components; to make the code faster and use fewer resources; to produce such configuration items as source files, content and test cases; software configuration management.

Exercise 6. *Read and translate text 1.*

Text 1. Software Construction Activities

The term software construction refers to the detailed creation of working, meaningful software through a combination of coding, verification, unit testing, integration testing, and debugging.

The software construction is linked to all the other software engineering activities, most strongly to software design and software testing. This is because the software construction process itself involves significant software design and test activity. It also uses the output of design and provides one of the inputs to testing. Detailed boundaries between design, construction, and testing (if any) will vary depending upon the software life cycle processes that are used in a project.

One of the key activities during construction is the integration of separately constructed routines, classes, components, and subsystems. In addition, a particular software system may need to be integrated with other software or hardware systems.

The following specific tasks are involved in software construction:

– verifying that the groundwork has been laid so that construction can proceed successfully;
– determining how your code will be tested;
– determining and writing classes and routines;
– creating and naming variables and named constants;
– selecting control structures and organizing blocks of statements;
– unit testing, integration testing, and debugging your own code;
– reviewing other team members' low-level designs and code and having them review yours;
– polishing the code by carefully formatting and commenting it;
– integrating the software components that were created separately;
– tuning the code to make it faster and use fewer resources.

Software construction typically produces configuration items that need to be managed in a software project (source files, content, test cases, and so on). Thus, the software construction is also closely linked to the software configuration management.

Exercise 7. *Find in text 1 the English for:*

конструювання програмного забезпечення; детальне створення робочого, змістовного програмного забезпечення; поєднання кодування, перевірки, тестування компонентів системи, тестування взаємодії компонентів системи та налагодження; бути пов'язаним з розробленням програмного забезпечення; проектування та тестування програмного забезпечення; чіткі межі; залежати від процесів життєвого циклу програмного забезпечення; поєднання підпрограм, класів, компонентів та підсистем; закласти основу; відбуватися

успішно; створення іменованих констант; вибір керівних конструкцій; створення блоків операторів; налагодження коду; перевірка низькорівневих програмних структур та коду; відрегулювати код так, щоб він був швидший та використовував менше ресурсів; керувати такими елементами конфігурації, як вихідні файли, зміст та набори тестових даних; керування конфігурацією програмного забезпечення.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. The term software construction refers to the detailed creation of working, meaningful software through a combination of coding, verification, unit testing, integration testing, and debugging. 2. The software construction process itself involves significant software design and test activity. 3. The software construction is linked to all the other software engineering activities, most strongly to software design and software modelling. 4. Detailed boundaries between design, construction, and testing (if any) will vary depending upon the hardware life cycle processes that are used in a project. 5. The software construction also uses the output of testing and provides one of the inputs to design. 6. One of the key activities during construction is the integration of separately constructed routines, classes, components, and subsystems. 7. Besides other things, software construction involves testing control structures and routines. 8. In addition, a particular software system may need to be integrated with other software or hardware systems. 9. The code can be polished by carefully formatting and commenting it. 10. Software construction typically produces configuration items that need to be managed in a software project (source files, content, test cases, and so on). 11. The code needs tuning to make it faster and use fewer resources. 12. The software construction is also closely linked to the project management.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to refer to	a) successfully
2) to depend on	b) tasks
3) to review	c) the creation of software
4) to be linked to	d) configuration items

5) to create	e) software engineering activities
6) to be integrated with	f) the software life cycle process
7) to test	g) low-level designs
8) to involve	h) software or hardware systems
9) to produce	i) named constants
10) to proceed	j) the code

Exercise 10. *Fill in the blanks with prepositions* ***in,(up)on, with, of, to, through, between*** *where necessary.*

1. The term software construction refers … the detailed creation … working, meaningful software … a combination … coding, verification, unit testing, integration testing, and debugging. 2. The software construction is linked … all the other software engineering activities, most strongly … software design and software testing. 3. It also uses the output … design and provides one … the inputs … testing. 4. Detailed boundaries … design, construction, and testing (if any) will vary depending … the software life cycle processes that are used … a project. 5. … addition, a particular software system may need to be integrated ... other software or hardware systems. 6. Some configuration items that need to be managed … a software project (source files, content, test cases, and so on).

Exercise 11. *Fill in the blanks with proper terms(****software engineering, debugging, source file, routine, verification, coding, software, software construction****) to complete the sentences.*

1. _____is establishment of the correctness of a theory, fact, activity, etc. 2. _____is the detailed creation of working, meaningful software through a combination of coding, verification, unit testing, integration testing, and debugging. 3. _____ is locating and removing defects in a device, system, plan, program, etc. 4. _____is a set of instructions, called a program, which tells a computer what to do. 5. _____ is writing texts of programs. 6. _____ is a part of a program performing a specific function. 7._____ is the original form of a program before it is converted into a machine-readable form. 8._____ is a systematic and disciplined approach to developing software which applies both computer science and engineering principles to the creation, operation and maintenance of the computer systems.

Exercise 12. *Answer the questions on text 1.*

1. What does the term software construction refer to? 2. What software engineering activities is the software construction linked to? 3. What do the detailed boundaries between design, construction, and testing depend on? 4. What is one of the key activities during construction? 5. What may a particular software system need to be integrated with? 6. What specific tasks are involved in construction? 7. How can the code be polished? 8. What is the purpose of code tuning? 9. What does software construction typically produce? 10. What is software construction also closely linked to?

Exercise 13. *Put all possible questions to the sentences below.*

1. The term software construction refers to the detailed creation of working, meaningful software. 2. The software construction is linked to all the other software engineering activities. 3. The software construction process itself involves significant software design and test activity. 4. Detailed boundaries between design, construction, and testing depend upon the software life cycle processes. 5. Software construction typically produces configuration items that need to be managed in a software project. 6. The software construction is also closely linked to the software configuration management.

Exercise 14. *Translate into English.*

1. Термін «конструювання програмного забезпечення» означає детальне створення робочого, змістовного програмного забезпечення шляхом поєднання кодування, перевірки, тестування компонентів системи, тестування взаємодії компонентів системи та налагодження її роботи. 2. Конструювання програмного забезпечення пов'язане з усіма іншими видами роботи з розроблення програмного забезпечення, найбільше з його проектуванням та тестуванням. 3. Сам процес конструювання програмного забезпечення передбачає багато роботи з проектування та тестування. 4. Також у процесі конструювання програмного забезпечення використовують вихідні дані проектування, які забезпечують один з вхідних параметрів тестування. 5. Чіткі межі між розробленням, конструюванням та тестуванням залежать від процесів життєвого циклу програмного забезпечення, що використовуються в проекті. 6. Одним з ключових

процесів конструювання є поєднання окремо створених підпрограм, класів, компонентів та підсистем. 7. Крім цього, окремі системи програмного забезпечення можуть потребувати поєднання з іншими програмними чи апаратними системами. 8. Спочатку необхідно перевірити, чи закладені основи для успішного конструювання. 9. Потім слід визначити спосіб тестування коду, написати класи та підпрограми, створити та назвати змінні та іменовані константи. 10. Після цього потрібно вибрати керівні конструкції та створити блоки операторів. 11. Всі члени команди повинні перевірити низькорівневі програмні структури та код один одного. 12. Код має бути налагоджений, «відшліфований» та відрегульований для його пришвидшення та використання меншої кількості ресурсів. 13. Під час конструювання програмного забезпечення зазвичай створюють елементи конфігурації, що потребують керування в програмному проекті. 14. Такими елементами конфігурації є вихідні файли, зміст та набори тестових даних. 15. Отже, конструювання програмного забезпечення також тісно пов'язане з керуванням конфігурацією програмних засобів.

Exercise 15. *Write a summary of the text "Software Construction" using the phrases:*

The text deals with the problem of …
- is devoted to …
- describes …
- focuses on …
- gives detailed information on …
- informs the readers of …

The key note of the text is …

Exercise 16. *Study the vocabulary to text 2.*

Identify – встановлювати, виявляти, визначати
numerous – численний
software architecture – архітектура програмного забезпечення
distinct – ясний, виразний; чіткий
definition – визначення, дефініція; тлумачення
maintenance – супровід, підтримка, експлуатація, технічне обслуговування
preexisting – який існував раніше; що існує

substantial – 1) істотний, важливий, значний; великий; 2) основний, головний
creativity – 1) творчі здібності, здатність створювати; 2) креативність, потенціал інформації
judgement – судження, думка, оцінка
interface – інтерфейс, взаємодія, взаємозв'язок
user-interface – користувацький інтерфейс
affect – впливати
ultimate – остаточний, кінцевий
call for – вимагати, потребувати

Exercise 17. *Read and translate text 2.*

Text 2. Software Development

Developing computer software can be a complicated process, and in the last 25 years researchers have identified numerous distinct activities that go into software development. They include:

- problem definition;
- requirement development;
- construction planning;
- software architecture or high-level design;
- detailed design;
- coding and debugging;
- unit testing;
- integration testing;
- integration;
- system testing;
- corrective maintenance.

These activities may be grouped together as "programming" or "creating a software product". If you create software on informal projects you deal with the activity the researchers refer to as "construction".

Construction focuses on coding and debugging but also includes detailed design, unit testing, integration testing, and other activities. Construction is also sometimes referred to as "coding" or "programming". But "coding" isn't really the best word because it implies the mechanical translation of preexisting design into a computer language. Construction is not at all mechanical and involves substantial creativity and judgement.

But what activities are not part of construction? Important non-constructional activities include management, requirements development, software architecture, user-interface design, system testing, and maintenance. Each of these activities affects the ultimate success of a project as much as construction – at least the success of any project that calls for more than one or two people and lasts longer than a few weeks.

Exercise 18. *Find in the text 2 the English for:*

складний процес; визначати багато окремих процесів; випрацювання вимог; визначення завдання; високорівневе проектування; кодування та налагодження; тестування компонентів системи та тестування взаємодії компонентів системи; коригувальний супровід; мати справу з роботою, яку дослідники називають конструюванням; потребувати значної креативності та розсудливості; проектування інтерфейсу користувача; кінцевий успіх проекту; передбачати участь більш як двох людей, продовжуватися більш як кілька тижнів.

Exercise 19. *Answer the questions on text 2.*

1. What activities does software development include? 2. What may these activities be named as? 3. What does software construction focus on? 4. What is construction sometimes referred to as? 5. Is construction just a mechanical process? 6. What activities are not a part of construction? 7. How do these activities affect the ultimate success of a project?

Exercise 20. *Use the proper tense form of the verbs in brackets. (Present, Past or Future Indefinite).*

1. Software (be) just instructions which (tell) the computer what to do. 2. The IBM 360 (be) the first commercially successful computer family. 3. The Internet (be) the biggest network in the world. 4. Computer professionals (decide) which hardware, software, and networks endure. 5. The PC (start) a revolution which affects nearly everything we do today. 6. Some common operating systems still in use (be) Windows 2000, Windows XP and Windows Vista. 7. Most software (change) over time and the anticipation of change (drive) many aspects of software construction. 8. Our former network administrator (relate) with routing technology such as Cisco. 9. Distance learning and videoconferencing

(be) concepts made possible with the use of an electronic classroom. 10. As computers (evolve) throughout the late 20th century, they (become) more interactive. 11. The objective functions (depend) on the perspective of the model's user. 12. The international company (store) their customer information in a central database in Brussels. 13. Today, computers in security systems (result) in safer environments. 14. Our homes and even objects on the street probably (interact) with our smartphones seamlessly. 15. Computer languages that (require) an interpreter often (run) slower than languages that (require) a compiler. 16. The system administrator (need) to upgrade the machine hardware. 17. The iPhone (have) all the features of a PDA, mobile phone, and an MP3 player in one package. 18. The evolution of nanocomputer technology (enable) us to build microscopic machines. 19. Operating system software (run) on laptop computers, cell phones and other so-called embedded devices. 20. During next ten years the size and shape of the computer (become) a major design issue. 21. Steve Jobs (be) famous for making high quality computers. 22. Computers can (help) people work more creatively. 23. The continued success of mainframes likely (depend) on the functionality. 24. Programs written in Java can (run) on many different computer architectures and operating systems. 25. The popularity of computer networks sharply (increase) with the creation of the World Wide Web (WWW) in the 1990s. 26. The Internet's technical changes (have) an increasing effect on our social and political structures. 27. The early versions of Microsoft Windows (not provide) any computer networking support. 28. With small computing devices people (be) able to spend more time doing what they often do best. 29. The specialist (use) Bluetooth technology to create a personal area network (PAN). 30. Without innovations in the areas of microprocessor and software reliability future systems (face) continuous failure.

Exercise 21. *Choose the right form of the verbs in brackets and translate the sentences. Mind the sequence of tenses.*

1. The IT support technician asked the end user how often he (update/is updating/updated) his device drivers. 2. The programmers know that PC servers (cost/will cost/costs) in the range of a few thousand dollars. 3. Analysts predict that our homes, cars and even objects on the street (interact/interacts/will interact) with our smart phones and with each other. 4. Teachers say that educational interaction (includes/was in-

cluded/will include) many factors, among them are students, curriculum, parents, teachers, administrators and more. 5. The woman said she always (is searching/searches/searched) for freeware versions of an application before buying one. 6. I'm sure the company (releases, will release, released) a new software next year. 7. The IT manager said the new information about architect's project (was/were/will be) unacceptable. 8. We know that online applications and services (will transform/ are transforming/transform) the consumer technology market. 9. The student asked his professor if the course books (was/were/are being) available as hypertext. 10. I suppose that the specialist (is knowing, knew, knows) several foreign languages.

Exercise 22. *Change the sentences into indirect speech.*

1. The lecturer told his students, "Microsoft Windows has a significant majority of market share in the notebook computer markets." 2. The professor warned our group, "Don't forget to review the material of the previous lecture on software engineering". 3. "You have computer problems that involve your operating system or an application", the IT specialist said to his client. 4. "Did you study any programming computer codes yesterday?" he asked his groupmates. 5. "Our analysts didn't establish computer security programs to prevent attacks", she complained. 6. My friend told me, "Tablets will take over from smartphones as the technology of choice for shopping". 7. "Last week we developed specialized databases for company needs," replied the programmer. 8. "What Internet provider will you choose?" asked him his colleague. 9. "Our technicians will help users deal with hardware and software problems", explained the support engineer. 10. I asked the networking specialist, "What kind of networks do you maintain?"

Exercise 23. *Use Present or Future Simple of the verbs in brackets. Mind that in subordinate clauses of time and condition, Present Simple is used instead of Future Simple. Such clauses begin with conjunctions: if, when, while, since, before, after, unless.*

1. After I (finish) school, I (enter) the University. 2. Since nobody (like) to wait for a computer, high-quality computers (have) fast processors and lots of quick memory. 3. Before she (get) to the theatre, she (go) past the computer centre. 4. If you (not have) previous experience, good

contacts, or a good degree from a well-known university, you (be) more successful in getting a lower-level job. 5. If students (keep) learning something new every day, they eventually (be) competent enough to get a high-skill job. 6. Hundreds of books (come) in the future as technologies (mature) and (evolve). 7. If you (work) on networks for a living and you (be) a network engineer, you probably (take) certification exams by networking companies such as Cisco. 8. Unless nanotechnology or some other technology actually (become) operational, this trend (end) according to some predictions around 2022. 9. While both mainframe and other platforms (evolve), there (be) some cost advantages to retaining the old technology. 10. When software (have) a bug the program (crash) and (terminate) with a confusing message.

Exercise 24. *Study the vocabulary to text 3.*

Fundamentals – основи
minimize – мінімізувати, зменшувати
complexity – складність
anticipate – передбачати
change – 1) зміна; 2) змінювати(ся)
concept – поняття, концепція
apply (*to*) – 1) стосуватися (*чогось*); 2) застосовувати(ся) (*у чомусь / до чогось*)
major – основний, головний
convey – передавати, висловлювати
intent – намір
severely – дуже, значно, досить
ability – можливість, здатність
driver – драйвер, *тут*: рушійна сила
essentially – по суті, посутньо, істотно, надзвичайно
particularly – дуже, надзвичайно, особливо
critical – важливий; необхідний
achieve – досягати
emphasize – надавати особливого значення, акцентувати увагу
drive – *тут:* запускати, керувати
unavoidably – неминуче
environment – оточення, середовище
in diverse ways – різними способами, інакше, по-різному
support – підтримувати, підтримка
technique – техніка, метод

communication method – метод переда(ва)ння інформації
operating system call – виклик операційної системи
tool – засіб, інструмент, сервісна програма
notation – система числення, система позначень, запис
Unified Modelling Language (UML)– уніфікована мова моделювання
fault – пошкодження, дефект, збій
ferret out – розшукувати, знаходити
as well as – так само, як; а також
software engineer – спеціаліст з розроблення програмного забезпечення
operational – операційний, оперативний
restricted – обмежений
issue – питання, проблема
interaction – взаємодія
software interface specification – специфікація, детальний опис програмного інтерфейсу
Object Management Group (OMG) – група керування об'єктами
International Organization for Standardization (ISO) – Міжнародна організація зі стандартизації

Exercise 25. *Translate the word combinations below into Ukrainian.*

Fundamentals of software construction; minimizing complexity; anticipating change; constructing for verification; to define the concepts; reduced complexity; to emphasize the creation of code; to be particularly critical to the process of verification and testing of software construction; to be a part of changing external environments; to affect software in diverse ways; to be supported by many specific techniques; communication method; programmer interface standards; operating system calls; to ferret out the faults; software engineers; to support code reviews; hard-to-understand language structures; standards that directly affect construction issues; numerous sources.

Exercise 26. *Read and translate text 3.*

Text 3. Software Construction Fundamentals

The fundamentals of software construction include:

- Minimizing complexity
- Anticipating change
- Constructing for verification
- Standards in construction

The first three concepts apply to design as well as to construction. We will define these concepts and describe how they apply to construction.

Minimizing Complexity

A major factor in how people convey intent to computers is the severely limited ability of people to hold complex structures and information in their working memories, especially over long periods of time. This leads to one of the strongest drivers in software construction: minimizing complexity. The need to reduce complexity applies to essentially every aspect of software construction, and is particularly critical to the process of verification and testing of software construction.

In software construction, reduced complexity is achieved through emphasizing the creation of the code that is simple and readable rather than clever.

Anticipating Change

Most software will change over time, and the anticipation of change drives many aspects of software construction. Software is unavoidably part of changing external environments, and changes in those outside environments affect software in diverse ways.

Anticipating change is supported by many specific techniques:

- Communication methods (for example, standards for document formats and contents)
- Programming languages (for example, language standards for languages like Java and C++)
- Platforms (for example, programmer interface standards for operating system calls)
- Tools (for example, diagrammatic standards for notations like UML (Unified Modelling Language))

Constructing for Verification

Constructing for verification means building software in such a way that faults can be ferreted out readily by the software engineers writing the software, as well as during independent testing and operational activities. Specific techniques that support constructing for verification include coding standards to support code reviews, unit testing, organizing code to support automated testing, and restricted use of complex or hard-to-understand language structures.

Standards in Construction

Standards that directly affect construction issues include the use of external standards. Construction depends on the use of external stan-

dards for construction languages, construction tools, technical interfaces, and interaction between software construction and other software engineering. Standards come from numerous sources, including hardware and software interface specifications such as the Object Management Group (OMG) and international organizations such as the ISO.

The Use of Internal Standards

Standards may also be created on an organizational basis at the corporate level or for use on specific projects. These standards support coordination of group activities, minimizing complexity, anticipating change, and constructing for verification.

Exercise 27. *Answer the questions on text 3.*

1. What do the fundamentals of software construction include? 2. What does the need to reduce complexity apply to? 3. What is reduced complexity achieved through? 4. What specific techniques is anticipating change supported by? 5. What does constructing for verification mean? 6. What do specific techniques that support constructing for verification include? 7. What does construction depend on? 8. What do standards in construction come from? 9. May standards also be created on an organizational basis?

Exercise 28. *Make up questions to the italicized parts of the sentences.*

1. The first three concepts apply to ***design as well as to construction***. 2. This leads to ***one of the strongest drivers in software construction***. 3. ***Reduced complexity*** is achieved through ***emphasizing the creation of code***. 4. ***Most software*** *will* ***change*** over time. 5. Anticipating change is supported by ***many specific*** techniques. 6. Standards ***that directly affect construction issues*** include ***the use of external standards.*** 7. ***Construction*** depends on ***the use of external standards***. 8. Standards come from ***numerous*** sources. 9. ***These standards*** support coordination of group activities, minimizing complexity, anticipating change, and constructing for verification.

Exercise 29. *Give nouns corresponding to the following verbs. Translate them.*

Construct, verify, apply, define, reduce, achieve, create, anticipate, communicate, program, operate, automate, restrict, direct, manage, interact, organize, specify, coordinate, act.

Exercise 30. *Study the vocabulary to text 4.*

Linear – лінійний
waterfall model – водоспадна модель
staged-delivery model – каскадна модель
treat – розглядати, трактувати, інтерпретувати
occur – траплятися, відбуватися
prerequisite – 1) передумова; 2) необхідний як передумова
complete – закінчувати, завершувати
requirement – вимога
tend to – мати тенденцію, схильність (*до чогось*)
precede – передувати
emphasis – акцент, наголос
iterative – ітеративний; той, що повторюється, повторюваний
prototyping – макетування, розроблення прототипу
extreme programming – екстремальне програмування
concurrently – одночасно, паралельно
overlap – перекривати(ся), частково збігатися
approach – підхід
consequently – тому, в результаті (*того, що*)
choice – вибір
extent – ступінь, міра, обсяг, величина
objective – мета
define – визначати
allocation – виділення (*ресурсу*), розташування, розподіл
assignment – призначення, присвоєння, розподіл

Exercise 31. *Translate the word combinations below into Ukrainian.*

Waterfall and staged-delivery life cycle models; to treat construction as an activity; significant prerequisite work has been completed; extensive design work; to emphasize the activities that precede construction (requirements and design); to create more distinct separations between the activities; to be more iterative, such as evolutionary prototyping and extreme programming; to occur concurrently with other software development activities; the approaches, which tend to mix design, coding, and testing activities; the choice of construction method; to affect the extent to which construction prerequisites are performed; to affect the project's ability to reduce complexity, anticipate change, and construct

for verification; to be addressed at the process, requirements, and design levels; to be influenced by the choice of construction method; to define the order in which components are created and integrated; the software quality management processes; the allocation of task assignments to specific software engineers.

Exercise 32. *Read and translate text 4.*

Text 4. Construction Models and Construction Planning

Numerous models have been created to develop software, some of which emphasize construction more than others.

Some models are more linear from the construction point of view, such as the waterfall and staged-delivery life cycle models. These models treat construction as an activity which occurs only after significant prerequisite work has been completed – including detailed requirements work, extensive design work, and detailed planning. The more linear approaches tend to emphasize the activities that precede construction (requirements and design), and tend to create more distinct separations between the activities. In these models, the main emphasis of construction may be coding.

Other models are more iterative, such as evolutionary prototyping and extreme programming. These approaches tend to treat construction as an activity that occurs concurrently with other software development activities, including requirements, design, and planning, or overlaps them. These approaches tend to mix design, coding, and testing activities, and they often treat the combination of activities as construction.

Consequently, what is considered to be "construction" depends to some degree on the life cycle model used.

The choice of construction method is a key aspect of the construction planning activity. The choice of construction method affects the extent to which construction prerequisites are performed, the order in which they are performed, and the degree to which they are expected to be completed before construction work begins.

The approach to construction affects the project's ability to reduce complexity, anticipate change, and construct for verification. Each of these objectives may also be addressed at the process, requirements, and design levels – but they will also be influenced by the choice of construction method.

Construction planning also defines the order in which components are created and integrated, the software quality management processes, the allocation of task assignments to specific software engineers, and the other tasks, according to the chosen method.

Exercise 33. *Find in text 4 the English for:*

численні моделі; розробляти програмне забезпечення; надавати особливого значення конструюванню; водоспадні та каскадні моделі життєвого циклу; розглядати конструювання як процес; важлива попередня робота; лінійний підхід; операції, що передують конструюванню; чітке розмежування; еволюційне моделювання; екстремальне програмування; поєднувати вимоги, проектування та планування; впливати на рівень виконання передумов конструювання; підхід до конструювання; вибір методу конструювання; порядок створення та поєднання компонентів; процес керування якістю програмного забезпечення; розподіл завдань.

Exercise 34. *Make five key questions to the text "Construction Models and Construction Planning".*

Exercise 35. *Study the vocabulary to text 5.*

Artifact – артефакт, продукт розроблення, робочий продукт
measure – 1) міра; 2) вимірювати, оцінювати, визначати
rate – 1) темп, швидкість, частота; 2) коефіцієнт; 3) інтенсивність
effort – зусилля; обсяг робіт, робота
schedule – 1) графік, розклад; план робіт; 2) планувати
fix – 1) визначати; 2) зафіксовувати, закріплювати
ensure – забезпечувати
come to terms (with) – домовлятися (*з кимось*); приймати (*чиїсь*) умови
arbitrary – довільний
chaotic – невпорядкований, хаотичний
constraint – обмеження
proximity – близькість; схожість
drive (*p.* drove, *pp.* driven) – рушати, приводити в рух, слугувати рушієм
consideration – міркування, підстава
craft-like – професійний, майстерний
explicitly – ясно, точно; відкрито, недвозначно

regardless of – не звертаючи уваги (*на що*), попри (*те, що*)
immovable – нерухомий, незмінний
impose (*a constraint*) – накладати (*обмеження*)
address – 1) спрямовувати (*зусилля*); 2) (to) звертатися (*до когось*); адресувати
scale – шкала, масштаб
small-scale – невеликий, незначний
account for – враховувати
unanticipated – непередбачуваний
gap – прогалина (*у чомусь*), брак (*чогось*)
flesh out – конкретизувати

Exercise 36. *Read and translate text 5.*

Text 5. Construction Measurement and Design

Numerous construction activities and artifacts can be measured, including code development, code modification, code reuse, code destruction, code complexity, code inspection statistics, fault-fix and fault-find rates, effort, and scheduling. These measurements can be useful for purposes of managing construction, ensuring quality during construction, improving the construction process, as well as for other reasons.

Construction is an activity in which the software has to come to terms with arbitrary and chaotic real-world constraints. Due to its proximity to these constraints, construction is more driven by practical considerations and software engineering is perhaps most craft-like in the construction area.

Some projects allocate more design activity to construction; others to a phase explicitly focused on design. Regardless of the exact allocation, some detailed design work will occur at the construction level, and that design work tends to be dictated by immovable constraints imposed by the real-world problem that is being addressed by the software. Just as construction workers building a physical structure must make small-scale modifications to account for unanticipated gaps in the builder's plans, software construction workers must make modifications on a smaller or larger scale to flesh out details of the software design during construction.

Exercise 37. *Find in text 5 the English for:*

оцінювати численні процеси та артефакти конструювання; включати розроблення, модифікацію, повторне використання, руйнування та складність коду; включати статистику перегляду коду, коефіцієнт визначення та знаходження помилки, обсяг робіт та планування; з метою керування конструюванням, забезпечення якості в процесі конструювання та вдосконалення конструювання; пристосовуватися до довільних та хаотичних обмежень реального світу; через подібність до обмежень реального світу; розроблення програмного забезпечення; бути найбільш майстерним у конструюванні; не звертаючи уваги на чіткий розподіл роботи; відбуватися на рівні конструювання; зумовлюватися незмінними обмеженнями завдань реального світу; вносити незначні зміни; конкретизувати деталі розроблення програмного забезпечення.

Exercise 38. *Write derivatives of the verbs below and explain their meanings.*

Measure, develop, use, modify, destruct, act, improve, consider, move, anticipate, allocate, require, restrict, specify, design.

Exercise 39. *Find in the text words that can function both as nouns and verbs. Translate them.*

Model: work – 1) праця
2) працювати

Exercise 40. *Answer the questions on text 5.*

1. What construction activities and artifacts can be measured? 2. What can these measurements be useful for? 3. What constraints does the software have to come to terms with? 4. What is construction driven by? 5. Why must software construction workers make modifications on a smaller or larger scale design?

Exercise 41. *Find some additional information and speak on:*

1. Software design.
2. Software testing.
3. Coding and debugging.

UNIT 2. SOFTWARE MODELLING

Exercise 1. *Study the basic vocabulary.*

a) terms

software model – модель програмного забезпечення
software modelling – моделювання програмного забезпечення
Unified Modelling Language (UML) – мова UML, уніфікована мова моделювання)
diagram – діаграма, схема
object-oriented design – об'єктно-орієнтоване проектування
legacy system – успадкована система *(система, що не задовольняє вимог, але використовується через складність її заміни)*
fix a bug – виправити помилку (усунути збій)
dependent quantity – залежна величина
process model – модель процесу
external perspective – зовнішній аспект
behavioural perspective – поведінковий аспект
structural perspective – структурний аспект
data architecture – архітектура (структура) даних

b) nouns

practice – 1) технологія, практика; 2) метод, спосіб
feature – властивість, характеристика, особливість
image – зображення
set – сукупність, набір
rule – правило
owner – власник
ownership – право власності
dependency – залежність
dependent – *тут:* залежна величина
identity – ідентичність
violation – порушення
cause – 1) причина; 2) спричиняти, викликати
destruction – руйнування, знищення
precedent – прецедент; *тут:* попереднє значення
behaviour – поведінка, режим роботи
communication – спілкування, зв'язок

stakeholder – учасник
perspective – ракурс, точка зору, аспект

c) verbs
conjure up – викликати в уяві
print – друкувати
start from scratch – розпочинати з нуля (з чистого аркуша)
permeate – проходити крізь, проникати
evaluate – оцінювати, визначати
reference – 1) давати посилання, посилатися (*на щось*); 2) подавати у вигляді таблиць
implement – реалізувати, здійснювати, забезпечувати
allow – дозволяти
interchange – 1) обмінювати(ся); 2) переставляти, міняти місцями
access – (отри)мати доступ
expect – очікувати
support – підтримувати

d) adjectives
wall-sized – розміром зі стіну
internal – внутрішній
out of date – застарілий
unwieldy – громіздкий
available – наявний, підхожий
entire – цілий, повний, увесь
responsible – відповідальний
transferable – який може передаватися
up to date – сучасний, оновлений
unique – унікальний, незвичайний
identical – однаковий, тотожний, ідентичний
uniform – сталий, рівний, однаковий

e) adverbs
fortunately – на щастя
probably – ймовірно
briefly – коротко, стисло
exactly – точно
whenever – кожного разу, коли; щоразу, коли б не
inherently – за своєю сутністю, у своїй основі; від природи

f) prepositions
unlike – на відміну від

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Permeate, however, modification, responsible, fortunately, object, inherently, conjure up, diagram, dependent, environment, unlike, probably, language, unwieldy, legacy, identify, inherently, interface, briefly, network, evaluate, whenever, generation.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) diagram, apply to, bug, architecture, create, permeate, implement, allow, support, feature, cause, perspective, environment, activity, wall-sized, unwieldy, identical, unique, uniform, quickly, fortunately, briefly, probably, apply;

b) dependent, destruction, out of date, internal, unwieldy, unique, responsible, quickly, fortunately, unlike.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

.

Model: create – creation – creator – creativity – creature – creative

Create, internal, fortune, simple, dependence, identify, violate, probable, brief, responsible, change, behaviour, differ, develop, architect, structure, apply, quick, cause, active.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

To conjure up images of wall-sized UML diagrams; diagrams are usually out of date by the time they are printed; large-scale and often unwieldy methods; to be not the only kind available for modelling software; to start from scratch; to permeate the entire system; to apply to all software models; ownership, dependency, interface and identity; to apply to legacy systems as well as to new projects; to think back to the last serious bug that you fixed; to identify a violation of one of these rules as the cause; responsible for creation and destruction; to be not

transferable; to be up to date whenever referenced; to implement some set of interfaces; to inherently identify an object; to access an identical object; to expect uniform behaviour; communication among stakeholders; to present the system from different perspectives; to show the system's context or environment; to show the system development process as well as activities supported by the system.

Exercise 6. *Read and translate text 1.*

Text 1. Basics of Software Modelling

Modelling is simply the practice of creating a small system that has some of the same features of a larger system. When applied to software, the word modelling usually conjures up images of wall-sized UML diagrams. Internal software, however, changes so quickly that such diagrams are usually out of date by the time they are printed. Fortunately, such large-scale and unwieldy methods are not the only kind available for modelling software.

A software model does not have to start from scratch. It does not have to permeate the entire system. The only thing that a software model needs is a set of rules. The same four rules apply to all software models.

The four rules of software modelling are:

1) ownership
2) dependency
3) interface
4) identity

Unlike the rules of object-oriented design, these four apply to legacy systems as well as to new projects. Think back to the last serious bug that you fixed. You can probably identify a violation of these rules as the cause.

Briefly, ownership means that every object has exactly one owner responsible for its creation and destruction, and that ownership is not transferable. Dependency means that any bit of data used to evaluate a dependent quantity is a precedent, and that dependents must be up to date whenever referenced. Interface means that every object implements some set of interfaces, and that these interfaces allow objects to be interchanged. Identity means that all objects have unique identity, that an interface inherently identifies an object, and that all clients accessing an identical object can expect uniform behaviour.

Software modelling helps the engineer to understand the functionality of the system. Models are used for communication among stakeholders. Different models present the system from different perspectives:

- external perspective showing the system's context or environment;
- process models showing the system development process as well as activities supported by the system;
- behavioural perspective showing the behaviour of the system;
- structural perspective showing the system or data architecture.

Exercise 7. *Find in text 1 the English for:*

щодо програмного забезпечення; викликати в уяві зображення величезних UML діаграм; застарілий; великомасштабні та громіздкі методи; розпочинатися з нуля; проходити через всю систему; набір правил; право власності, залежність, взаємозв'язок та ідентичність; на відміну від правил об'єктно-орієнтованого проектування; застосовуватися до успадкованих систем так само, як і до нових проектів; виправляти серйозну помилку; порушення правил; створення та знищення; право власності не передається; визначати залежну величину; бути найновішими щоразу, коли до них звертаються; забезпечувати певну сукупність інтерфейсів; дозволяти переставляти об'єкти; мати доступ до ідентичного об'єкту; очікувати однакової поведінки; обмін інформацією між учасниками; презентувати систему з різних ракурсів; поведінка системи.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. When applied to software, the word modelling usually conjures up images of matchbox-sized circuit diagrams. 2. Modelling is simply the practice of creating a large system that has some of the same features of a smaller system. 3. Unfortunately, large-scale and unwieldy methods are the only kind available for modelling software. 4. Internal software does not change quickly and diagrams are usually up to date by the time they are printed. 5. A software model has to permeate the entire system. 6. A software model does not have to start from scratch. 7. The four rules of software modelling are: significance, availability, simplicity and minimization. 8. Unlike the rules of object-oriented design, these four do not apply to legacy systems as they do to new projects.

9. Dependency means that any bit of data used to evaluate a dependent quantity is a precedent, and that dependents must be up to date whenever referenced. 10. Briefly, ownership means that every object has exactly one owner responsible for its creation and destruction. 11. Interface means that every object implements some set of interfaces, and that these interfaces do not allow objects to be complementary to one another. 12. Software modelling helps the engineer to understand the functionality of the system. 13. Identity means that all objects have unique identity, that an interface inherently identifies an object, and that all clients accessing an identical object can expect uniform behaviour. 14. Different models present the system from different perspectives: external perspective, system development perspective, behavioural perspective and structural perspective.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1. to change	a) uniform behaviour
2. to be	b) the system from different perspectives
3. to start from	c) the last serious bug
4. to apply to	d) communication among stakeholders
5. to think back to	e) legacy systems
6. to evaluate	f) some set of interfaces
7. to implement	g) scratch
8. to expect	h) a dependent quantity
9. to be used for	i) quickly
10. to present	j) out of date

Exercise 10. *Fill in the blanks with prepositions* ***to, of, by, from, unlike, with, among, up, for*** *where necessary.*

1. Modelling is simply the practice … creating a small system that has some … the same features … a larger system. 2. When applied … software, the word modelling usually conjures … images … wall-sized UML diagrams. 3. Internal software, however, changes so quickly that such diagrams are usually … … date … the time they are printed. 4. A software model does not have to start … scratch. 5. … the rules … object-oriented design, these four apply … legacy systems as well as … new projects. 6. Briefly, ownership means that every object has exactly one owner responsible … its creation and destruction. 7. Software

modelling helps the engineer to understand the functionality … the system. 8. Models are used … communication … stakeholders. 9. A software model has to permeate … the entire system. 10. Different models present the system … different perspectives.

Exercise 11. *Fill in the blanks with proper terms (**structural perspective, behavioural perspective, process models, external perspective, modelling, identity, interface, dependency, ownership**) to complete the sentences.*

1._____ means that every object implements some set of interfaces, and that these interfaces allow objects to be interchanged. 2. _____is simply the practice of creating a small system that has some of the same features of a larger system. 3._____ shows the system's context or environment. 4._____ means that any bit of data used to evaluate a dependent quantity is a precedent, and that dependents must be up to date whenever referenced. 5._____show the system development process as well as activities supported by the system. 6._____ means that every object has exactly one owner responsible for its creation and destruction, and that ownership is not transferable. 7. _____shows the system or data architecture. 8._____shows the behaviour of the system. 9._____means that all objects have unique identity, that an interface inherently identifies an object, and that all clients accessing an identical object can expect uniform behaviour.

Exercise 12. *Answer the questions on text 1.*

1. What is modelling? 2. What does the word modelling usually conjure up when applied to software? 3. Why are such diagrams usually out of date by the time they are printed? 4. Does a software model have to start from scratch or permeate the entire system? 5. What four rules apply to all software models? 6. What projects do these four rules apply to? 7. What can violation of these rules lead to? 8. What does ownership mean? 9. What does dependency mean? 10. What does interface mean? 11. What does identity mean? 12. What does software modelling help the engineer in? 13. What perspectives do different models present the system from? 14. What do these perspectives show?

Exercise 13. *Put all possible questions to the sentences below.*

1. The word modelling usually conjures up images of wall-sized UML diagrams. 2. Internal software changes quickly. 3. Such diagrams are usually out of date by the time they are printed. 4. The only thing that a software model needs is a set of rules. 5. The same four rules apply to all software models.

Exercise 14. *Translate into English.*

1. Моделювання – це метод створення меншої системи, що має певні характеристики більшої. 2. Щодо програмного забезпечення слово «моделювання» зазвичай викликає в уяві зображення UML діаграм розміром зі стіну. 3. Проте вміст програмного забезпечення змінюється так швидко, що діаграми зазвичай стають застарілими раніш, як їх роздрукують. 4. На щастя, для моделювання програмного забезпечення існують не лише такі великомасштабні та громіздкі методи. 5. Модель програмного забезпечення необов'язково має розпочинатися з нуля й проходити через усю систему. 6. Єдине, чого потребує модель програмного забезпечення – це набір правил. 7. Чотири правила моделювання програмного забезпечення – це право власності, залежність, взаємозв'язок та ідентичність. 8. На відміну від правил об'єктно-орієнтованого проектування, ці чотири правила застосовують до успадкованих систем так само, як і до нових проектів. 9. Згадайте останню серйозну помилку, яку ви виправили. 10. Причиною цього, ймовірно, можна вважати порушення цих правил. 11. Право власності означає, що кожний об'єкт має лише одного власника, і право власності не передається. 12. Залежність означає, що будь-яка кількість даних, які використовуються для визначення залежної величини, є попереднім значенням. 13. Взаємозв'язок означає, що кожний об'єкт забезпечує певну сукупність інтерфейсів, які дозволяють переставляти об'єкти. 14. Ідентичність означає, що всі об'єкти мають унікальну ідентичність, і інтерфейс, що по суті, визначає об'єкт. 15. Моделювання програмного забезпечення допомагає інженерові зрозуміти функціональність системи. 16. Різні моделі представляють систему з різних аспектів. 17. Зовнішній аспект показує контекст або оточення системи. 18. Моделі процесу показують процес розроблення системи, а також функції, які підтримує ця система. 19. Поведінковий аспект показує поведінку системи. 20. Структурний аспект показує архітектуру системи або даних.

Exercise 15. *Write a summary of the text "Software Modelling".*

Exercise 16. *Study the vocabulary to text 2.*

Natural science – природознавство; одна з природничих наук (*фізика, хімія*)
engineering discipline – інженерна / технічна дисципліна
meteorology – метеорологія
electrical engineering – електротехніка
social sciences – суспільні науки
political science – політологія
usable – придатний для використання, практичний, зручний
static model – статична модель
differential equation – диференційне рівняння
game theory – теорія ігор
overlap – перекривати(ся), частково збігатися
optimize – оптимізувати
hypothesis – гіпотеза
estimate – оцінювати, підраховувати
unforeseeable event – непередбачувана подія
affect – впливати
similarly – подібним чином
simulation – моделювання
equation – рівняння
relation(ship) – (взаємо)зв'язок, залежність, співвідношення
real number – дійсне число
integer number – ціле число
boolean – булівський, булів
string – рядок
property – властивість
timing data – часові показники / характеристики
counters – лічильні функції
event occurrence – настання події
decision variable – змінна рішення
input variable – вхідна змінна
state variable – змінна стану
exogenous variable – екзогенна (*зовнішня*) змінна

random variable – випадкова змінна
output variable – вихідна змінна
constant – постійна величина, константа
furthermore – до того ж, крім того, більш того
objective – мета
constraint – обмеження
objective function – цільова функція
perspective – *тут:* бачення, концепція, точка зору
index of performance – показник ефективності / продуктивності
measure – 1) показник, критерій; 2) вимірювати
involved – складний, заплутаний (*механізм*)

Exercise 17. *Read and translate text 2.*

Text 2. Mathematical Models

A mathematical model uses mathematical language to describe a system. Mathematical models are used not only in the natural sciences and engineering disciplines (such as physics, biology, meteorology, and electrical engineering) but also in the social sciences (such as economics, psychology, sociology and political science); physicists, engineers, computer scientists, and economists use mathematical models most extensively.

Eykhoff (1974) defined a mathematical model as "a representation of the essential aspects of an existing system (or a system to be constructed) which presents knowledge of that system in usable form".

Mathematical models can take many forms, including but not limited to dynamic systems, static models, differential equations, or game theoretic models. These and other types of models can overlap, with a given model involving a variety of abstract structures.

Often when engineers analyze a system to be controlled or optimized, they use a mathematical model. In analysis, engineers can build a descriptive model of the system as a hypothesis of how the system could work, or try to estimate how an unforeseeable event could affect the system. Similarly, in control of a system, engineers can try out different control approaches in simulations.

A mathematical model usually describes a system by a set of variables and a set of equations that establish relationships between the variables. The values of the variables can be practically anything; real or

integer numbers, boolean values or strings, for example. The variables represent some properties of the system, for example, measured system outputs often in the form of signals, timing data, counters, and event occurrence (yes/no). The actual model is the set of functions that describe the relations between the different variables.

There are six basic groups of variables: decision variables, input variables, state variables, exogenous variables, random variables, and output variables. Since there can be many variables of each type, the variables are generally represented by vectors.

Decision variables are sometimes known as independent variables. Exogenous variables are sometimes known as parameters or constants. The variables are not independent of each other as the state variables are dependent on the decision, input, random, and exogenous variables. Furthermore, the output variables are dependent on the state of the system (represented by the state variables).

Objectives and constraints of the system and its users can be represented as functions of the output variables or state variables. The objective functions will depend on the perspective of the model's user. Depending on the context, an objective function is also known as an index of performance, as it is some measure of interest to the user. Although there is no limit to the number of objective functions and constraints a model can have, using or optimizing the model becomes more involved (computationally).

Exercise 18. *Make up 10 key questions on the text.*

Exercise 19. *Find in text 2 the English for:*

природничі науки та технічні дисципліни; соціологія та політологія; метеорологія та електротехніка; найбільш широко використовувати математичні моделі; представлення важливих аспектів наявної системи; презентувати знання про систему в практичній формі; набувати багатьох форм; включно з динамічними системами, але не обмежуючись ними, статичними моделями, диференційними рівняннями або моделями теорії ігор; моделі можуть накладатися одна на одну; включати різноманітні абстрактні структури; описова модель системи; оцінювати, як непередбачувана подія може вплинути на систему; випробовувати різні концепції керування в моделюванні; описувати систему сукупністю змінних та сукупністю рівнянь; встановлювати зв'язок між змінними; дійсні або цілі числа; булеві

величини або рядки; часові характеристики, лічильні функції та настання події; змінні рішення, вхідні змінні, змінні стану, екзогенні (зовнішні) змінні, випадкові змінні та вихідні змінні; цілі та обмеження системи; цільова функція, відома як показник ефективності; показник, що цікавить користувача; моделі використання та оптимізування стають дедалі складнішими.

Exercise 20. *Study the vocabulary to text 3.*

Vs – проти, відносно, замість
quantity – величина, параметр
otherwise – в іншому випадку, інакше
linearity – лінійність
linearization – лінеаризація
assume – припускати, вважати
predictor variable – предикторна змінна, прогностичний параметр
fairly – досить, доволі
associate – пов'язувати
irreversibility – незворотність
deterministic – детермінувальний; визначальний
probabilistic (*stochastic*) model – і/ймовірнісна (*стохастична*) модель
state – стан
be uniquely determined – однозначно визначатися
conversely – навпаки
randomness – випадковість, і/ймовірність
unique value – єдине значення
probability distribution – розподіл імовірностей
lumped parameter – зосереджений параметр
distributed parameter – розподілений параметр
homogeneous – однорідний, гомогенний
consistent – *зуст:* однаковий, єдиний, стабільний
heterogeneous – гетерогенний, неоднорідний

Exercise 21. *Translate the word combinations below into Ukrainian.*

Linear vs nonlinear; predictor variables, differential equation, objective functions and constraints; in fairly simple systems; to be often associated with phenomena such as chaos and irreversibility; a common approach to nonlinear problems; aspects such as irreversibility, which are

strongly tied to nonlinearity; deterministic vs probabilistic (stochastic) model; every set of variable states; to be uniquely determined by parameters; unique value; probability distribution; lumped vs distributed parameters; consistent state throughout the entire system; varying state within the system.

Exercise 22. *Read and translate text 3.*

Text 3. Classification of Mathematical Models

Many mathematical models can be classified in the following ways:

Linear vs nonlinear. Mathematical models are usually composed of variables, which are abstractions of quantities of interest in the described systems, and operators that act on these variables, which can be algebraic operators, functions, differential operators, etc. If all the operators in a mathematical model present linearity, the resulting mathematical model is defined as linear. A model is considered to be nonlinear otherwise.

The question of linearity and nonlinearity is dependent on the context, and linear models may have nonlinear expressions in them. For example, in a static linear model, it is assumed that a relationship is linear in the parameters, but it may be nonlinear in the predictor variables. Similarly, a differential equation is said to be linear if it can be written with linear differential operators, but it can still have nonlinear expressions in it. In a mathematical programming model, if the objective functions and constraints are represented entirely by linear equations, then the model is regarded as a linear one. If one or more of the objective functions or constraints are represented with a nonlinear equation, then the model is known as a nonlinear one.

Nonlinearity, even in fairly simple systems, is often associated with phenomena such as chaos and irreversibility. Although there are exceptions, nonlinear systems and models tend to be more difficult to study than linear ones. A common approach to nonlinear problems is linearization, but this can be problematic if one is trying to study aspects such as irreversibility, which are strongly tied to nonlinearity.

Deterministic vs probabilistic (stochastic). A deterministic model is one in which every set of variable states is uniquely determined by parameters in the model and by sets of previous states of these variables. Therefore, deterministic models perform the same way for a given set of initial conditions. Conversely, in a stochastic model, randomness is pre-

sent, and variable states are not described by unique values, but rather by probability distributions.

Static vs dynamic. A static model does not account for the element of time, while a dynamic model does. Dynamic models typically are represented with differential equations.

Lumped vs distributed parameters. If the model is homogeneous (consistent state throughout the entire system) the parameters are distributed. If the model is heterogeneous (varying state within the system), then the parameters are lumped. Distributed parameters are typically represented with partial differential equations.

Exercise 23. *Find in text 3 the English for:*

величини, що нас цікавлять; оператори, що діють на ці змінні; бути лінійним; питання лінійності та нелінійності; бути залежним від контексту; припускати, що; прогностичний параметр; диференційне рівняння; цільові функції та обмеження; бути представленим виключно лінійними рівняннями; досить прості системи; бути пов'язаним з такими явищами, як хаос та незворотність; загальний підхід; набір (сукупність) змінних; однозначно визначатися параметрами; зосереджені та розподілені параметри; однорідна модель; неоднорідна модель; стабільний стан; мінливий стан.

Exercise 24. *Translate into English.*

1. Математичні моделі складаються зі змінних та операторів, що впливають на ці змінні. 2. Змінні величини є абстракціями величин, які нас цікавлять у системах, що ми описуємо. 3. Операторами можуть бути алгебраїчні оператори, функції та диференційні оператори. 4. Якщо всі оператори в математичній моделі є лінійними, то кінцеву математичну модель визначають як лінійну. 5. В іншому випадку модель вважають нелінійною. 6. Питання лінійності та нелінійності залежить від контексту, і лінійні моделі можуть містити нелінійні вирази. 7. Нелінійність часто пов'язують з такими явищами, як хаос та незворотність. 8. Загальним підходом до нелінійних задач є лінеаризація. 9. Детермінована модель – це така, в якій кожна сукупність станів змінних однозначно визначається параметрами самої моделі та сукупностями попередніх станів цих змінних. 10. Тому детерміновані моделі діють однаково на даний набір

початкових умов. 11. В стохастичній моделі, навпаки, присутня випадковість, і стани змінних описуються не єдиними значеннями, а розподілом імовірності. 12. Статична модель не враховує елемент часу, тоді як динамічна його враховує. 13. Динамічні моделі зазвичай презентують диференційними рівняннями. 14. Якщо модель однорідна (стабільний стан у всій системі), то параметри є розподіленими. 15. Якщо модель є неоднорідною (мінливий стан в системі), то параметри є зосередженими.

Exercise 25. *Use the proper tense form of the verbs in brackets (Present, Past or Future Continuous).*

1. New types of computers constantly (come) out. 2. I (learn) new software packages at five o'clock yesterday. 3. This time next week, the technicians (troubleshoot) computer problems within a company. 4. Creating new pages for the Web (get) easier all the time. 5. Hardware engineer (design) communications devices for corporate use at the time. 6. This time tomorrow my colleague (work) for a famous hardware manufacturer such as Apple. 7. Computers (get) more powerful over previous generations. 8. When PC specialists (install) computers, they (give) some useful recommendations. 9. I wonder what my friend (do) this time next month. 10. Right now you most likely (use) a GUI interface. 11. The engineer (develop) software for the insurance company all morning yesterday. 12. I expect I still (work) with the same team of specialists. 13. Trends in computer storage constantly (change). 14. Touch screens rapidly (replace) keypads on mobile phones. 15. What you (do) next weekend? – I will (write) users manuals and training materials as usual. 16. Your network administrators (maintain) company's Internet connections while we (install) security services in our main office? 17. Computing equipment (get) smaller and more sophisticated. 18. I (demonstrate) multimedia products for our new customers at 9 o'clock tonight. 19. Our database analyst (train) our employees on the systems all day yesterday. 20. Diskettes (get) rare these days and they are replaced by USB flash memory drives. 21. I (take) part in the conference for network professionals tomorrow morning. 22. What network security analyst (do) when hackers attacked and damaged the computer system? 23. I (discuss) a new Web project with programmers when you see me next. 24. As the specialist (design) for University's website he (use) different sets of programming languages.

25. Look at the time. Your client (come) in a minute and you haven't even started testing a new computer equipment. 26. Digital distribution over a network rapidly (replace) removable storage media such as CD-ROM's and DVD-ROM's. 27. This time next week he (buy) new hardware. 28. When I first met IT professional he (replace) hardware systems. 29. I am afraid our professor (not deliver) lectures on network design this time next term. 30. Where he (create) multimedia products when you saw him last?

Exercise 26. *Complete the sentences below using the appropriate tense forms of the verbs in brackets (Present /Past Indefinite or Present/Past Continuous).*

1. My computer exam (take place) next week. 2. When I (arrive) the professor (explain) the multi-leveled architecture of a large software system. 3. Dean (conduct) a survey at the moment investigating how students use free educational content on our University website. 4. He (download) e-mail files when malicious computer viruses (attack) and (disable) the antivirus application. 5. I (wonder) what these programmers (talk) about. – They (discuss) common computer problems and their solutions. 6. Network administrator (maintain) a computer network while other specialist (monitor) its security. 7. Software engineers (not take part) in program design today because the project is not approved. 8. I (read) my electronic course-book when suddenly the power (switch off). 9. What the student (do) to his computer system now? – I (think) he (install) an application program. 10. My colleague still (use) Windows XP when Microsoft (release) a new version with advanced features. 11. The programmer (design) new applications for business operations yesterday morning. 12. My friend usually (learn) foreign languages very quickly and now he (learn) German. 13. I just (update) the database at the time when I suddenly (note) unauthorized access. 14. Using different search engines (become) increasingly popular. 15. Why you (lend) him this book? I still (read) it.

Exercise 27. *Choose the right form of the verbs in brackets and translate the sentences. Mind the sequence of tenses.*

1. The programming editor said, "I (am writing, was writing, will be writing, write) the source code of an application". 2. The student asked

the lecturer what questions he (consider, would consider, was considering) in this lecture. 3. The clerk asked if she (typed, types, is typing, was typing) data into a database. 4. He wondered if employees (was using, use, will use, were using) packaged software. 5. Ann said with regret that she (can't, is able to, couldn't, will be able) install security programs. 6. He told us that he (is going, was going, will go) to take an English exam the following year. 7. They informed that participants of the conference (discussed, will discuss, were discussing) the latest scientific discoveries at that moment.

Exercise 28. *Change the sentences into indirect speech.*

1. The lecturer said, "Current chip technology is approaching the limits of physics." 2. The teacher asked his students, "Are recent advances in computing communications and software transforming the way people live?" 3. "A number of changes are affecting mainframe technology," he said. 4. "What security programs were you testing when I saw you yesterday morning?" the programmer asked his colleagues. 5. "Are data analytics going to prevent new research opportunities to the computer graphics?" I asked the web designer. 6."The programmer was not using presentation software for his report yesterday morning", the lecturer said. 7. "Is the Internet of Things moving to the mainstream activity at the moment?" his friend asked him. 8. "The researchers were doing quantum computing experiments when I saw them last", the scientist claimed. 9. "In-memory computing is rapidly growing because of its power, versatility and incorporation into several software and hardware products," explained the hardware specialist. 10. The information manager told me, "Our company is planning to address big data in our integration infrastructure."

Exercise 29. *Study the vocabulary to text 4.*

Collage – колаж, комбінування різних елементів
general-purpose – універсальний, загального призначення
specify – визначати, задавати
visualize – наочно презентувати, візуалізувати
software intensive system – система з громіздким ПЗ
blueprint – ескіз, креслення, проект
actor – актор *(в мові UML – людина або пристрій, що взаємодіє з системою; його зображають у вигляді фігурки людини)*

database schema – схема даних, логічна структура даних *(зовнішній опис або діаграма заданої у СКБД структури запису; термін був запроваджений у 1971 р. для дворівневого підходу до опису структури БД)*
entity relationship diagram – діаграма відношень логічних об'єктів-сутностей, ER-діаграма
business modelling – бізнес-моделювання
workflow – послідовність операцій
throughout – впродовж
software development life cycle – життєвий цикл розроблення ПЗ
synthesize – синтезувати
notation – система позначень
Object-modellingtechnique (OMT) – метод об'єктного моделювання
Object-oriented software engineering (OOSE) – об'єктно-орієнтована програмотехніка
fuse – комбінувати, поєднувати
aim – 1) мета, намір, прагнення 2) прагнути
distributed system – розподілена система
de facto – фактично, де-факто
evolve – розвиватися
auspice – сприяння
customization – пристосування під вимоги замовника
profile – профіль, сукупність параметрів користувача
transformation – перетворення, трансформація
stereotype – стереотип *(у мові UML – створення нових елементів моделі шляхом розширення функціональності базових елементів)*
compatible – сумісний, сполучний, схожий
recast – змінювати, переробляти
take advantage of – скористатися, використовувати
Rational Unified Process (RUP) – раціональний уніфікований процес *(розроблення)*

Exercise 30. *Translate the word combinations below into Ukrainian.*

Standardized general-purpose modelling language; to specify, visualize, modify, construct and document the artifacts; object-oriented software intensive system under development; to visualize a system's architectural blueprints; actors, business processes, (logical) components, activi-

ties, programming language statements, database schemas and reusable software components; best techniques of data modelling (entity relationship diagrams); business modelling (workflows); object modelling and component modelling; throughout the software development life cycle; to synthesize the notations of the Object-modelling technique (OMT) and Object-oriented software engineering (OOSE); common and widely usable modelling language; to model concurrent and distributed systems; a de facto industry standard; to evolve under the auspices of the Object Management Group (OMG); to be extensible; mechanisms for customization; to be compatible (with); to recast the methods; to take advantage of the new notations; Rational Unified Process (RUP); Abstraction Method; Dynamic Systems Development Method; to achieve different objectives.

Exercise 31. *Read and translate text 4.*

Text 4. Unified Modelling Language

Unified Modelling Language (UML) is a standardized general-purpose modelling language in the field of software engineering. It is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software intensive system under development. It offers a standard way to visualize a system's architectural blueprints, including elements such as actors, business processes, (logical) components, activities, programming language statements, database schemas and reusable software components.

UML combines best techniques of data modelling (entity relationship diagrams), business modelling (workflows), object modelling, and component modelling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies. UML has synthesized the notations of the Object-modelling technique (OMT) and Object-oriented software engineering (OOSE) by fusing them into a single, common and widely usable modelling language. It aims to be a standard modelling language which can model concurrent and distributed systems. It is a de facto industry standard, and is evolving under the auspices of the Object Management Group (OMG). UML models may be automatically transformed to other representations (e.g. Java) by means of transformation languages, supported by the OMG. UML is extensible, offering such mechanisms for customization as profiles and stereotypes.

UML is not a development method by itself, however, it was designed to be compatible with the leading object-oriented software development methods. Since UML has evolved, some of these methods have been recast to take advantage of the new notations, and new methods have been created based on UML. The best known is IBM Rational Unified Process (RUP). There are many other UML-based methods like Abstraction Method, Dynamic Systems Development Method, and others, designed to provide more specific solutions, or achieve different objectives.

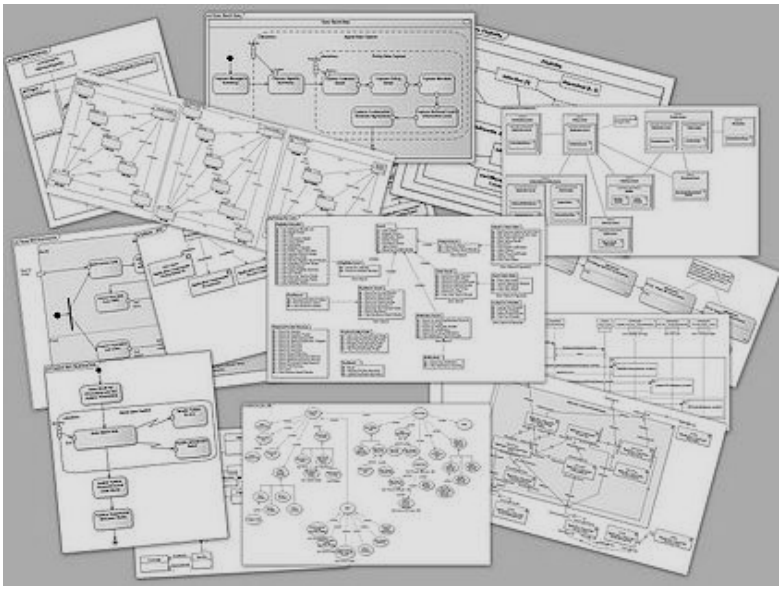

Fig. 1. A collage of UML diagrams

Exercise 32. *Answer the questions on text 4.*

1. What is UML? 2. What is UML used for? 3. In what way does UML visualize a system's architectural blueprints? 4. What kinds of modelling techniques does UML combine? 5. What notations has UML synthesized? 6. What kinds of systems can UML model? 7. What mechanisms does UML offer for customization? 8. Is UML a development method by itself? 9. What are the best known object-oriented software development methods?

Exercise 33. *Decipher the abbreviations below.*

UML, OMT, OOSE, OMG, RUP, ISO.

Exercise 34. *Study the vocabulary to text 5.*

Distinguish – відрізняти
partial – частковий
semantic backplane – семантичний задній план *(в UML об'єднує модель і наповнює її змістом)*
use case – прецедент *(у мові UML – з допомогою прецедентів моделюють діалог між актором і системою; набір усіх прецедентів системи визначає її функціональність; на діаграмах прецедент зображають у вигляді еліпса)*
view – 1) вид 2) погляд, аспект, точка зору
system mode – системний режим
emphasize – надавати особливого значення (*чомусь*), підкреслювати (*щось*), робити акцент, акцентувати увагу, наголошувати (*на чомусь*)
attribute – атрибут
composite structure diagram – діаграма композитних структур
collaboration – співпраця, спільна праця
sequence diagram – діаграма послідовностей
activity diagram – діаграма діяльності *(у UML – діаграма, на якій презентовані переходи потоку керування від однієї діяльності до іншої)*
state machine diagram – діаграма кінцевого автомату
hierarchically – ієрархічно
structure diagram – структурна діаграма
behaviour diagram – діаграма поведінки
class diagram – діаграма класів *(діаграма UML, яка презентує статичний погляд на систему з погляду класів і відношень між ними)*
component diagram – діаграма компонентів
object diagram – діаграма об'єктів
profile diagram – профілограма
deployment diagram – діаграма розгортання *(діаграма UML, яка окреслює фізичну конфігурацію системи в термінах вузлів і з'єднань між ними, наприклад, з допомогою обчислювальної мережі)*
package diagram – діаграма пакетів

use case diagram – діаграма прецедентів *(у мові UML – графічне зображення акторів, прецедентів та їх взаємодій у системі; розрізняють головну діаграму прецедентів і додаткові діаграми)*

interaction diagram – діаграма взаємодії *(загальна назва діаграм UML, на яких презентований динамічний погляд на систему з погляду об'єктів і повідомлень, якими вони обмінюються; практично використовується або діаграма кооперації, або діаграма послідовності)*

communication diagram – діаграма комунікації *(починаючи з UML 2.0*; *у UML 1x – це діаграма кооперації)*

interaction overview diagram – діаграма огляду взаємодії

timing diagram – часова діаграма

restrict – обмежувати

flexibility – гнучкість

extend – розширювати, збільшувати, нарощувати

engineering drawing – технічне креслення

intent – намір, мета

Exercise 35. *Translate the word combinations below into Ukrainian.*

To distinguish between the UML model and the set of diagrams of a system; a partial graphical representation of a system's model; to contain a "semantic backplane"; written use cases that drive the model elements and diagrams; different views of a system mode; static (or structural) view and dynamic (or behavioural) view; class diagrams and composite structure diagrams; to emphasize the dynamic behaviour of the system; to show collaboration among objects; sequence diagrams, activity diagrams and state machine diagrams; partially restricted flexibility; structure diagram and behaviour diagram; component diagram and object diagram; profile diagram, deployment diagram and package diagram; use case diagram and interaction diagram; communication diagram and timing diagram; interaction overview diagram.

Exercise 36. *Read and translate text 5.*

Text 5. UML Modelling

It is very important to distinguish between the UML model and the set of diagrams of a system. A diagram is a partial graphical representation of a system's model. The model also contains a

"semantic backplane" – documentation such as written use cases that drive the model elements and diagrams.

UML diagrams represent two different views of a system mode.

Static (or structural) view emphasizes the static structure of the system using objects, attributes, operations and relationships. The structural view includes class diagrams and composite structure diagrams.

Dynamic (or behavioural) view emphasizes the dynamic behaviour of the system by showing collaborations among objects and changes to the internal states of objects. This view includes sequence diagrams, activity diagrams and state machine diagrams.

UML 2.2 has 14 types of diagrams divided into two categories. Seven diagram types represent *structural* information, and the other seven represent general types of *behaviour*, including four that represent different aspects of *interactions*. These diagrams can be categorized hierarchically as shown in the class diagram (Fig. 2).

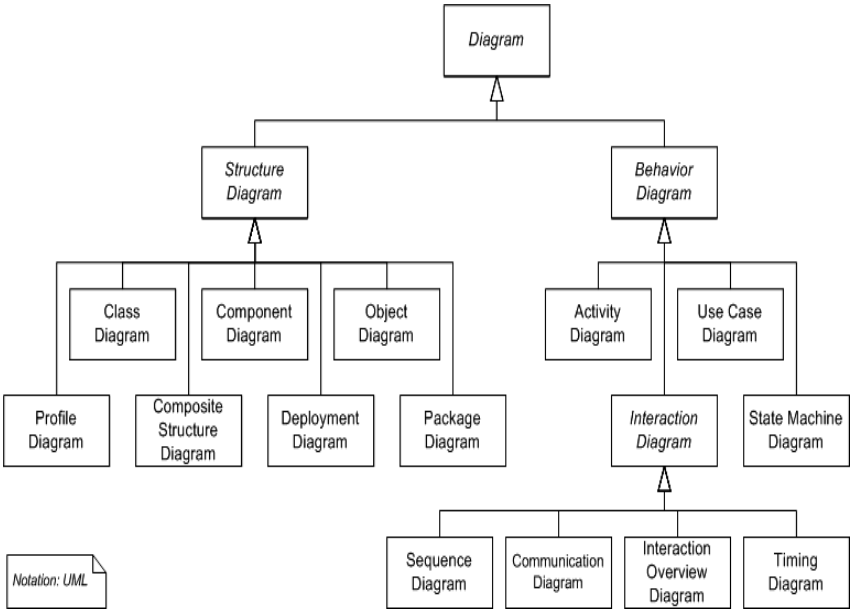

Fig. 2. UML class diagram

UML does not restrict UML element types to a certain diagram type. In general, every UML element may appear on almost all types of diagrams; this flexibility has been partially restricted in UML 2.0. UML

profiles may define additional diagram types or extend existing diagrams with additional notations.

In keeping with the tradition of engineering drawings, a comment or note explaining usage, constraint, or intent are allowed in a UML diagram.

Exercise 37. *Do the assignments below.*

1. Explain the difference between a model and a diagram.
2. Compare two different views of a system mode.
3. Name the categories of UML 2.2 diagrams.
4. Name the diagrams which represent each category.

Exercise 38. *Study the vocabulary to text 6.*

Depict – описувати, зображувати
split up – розподіляти(ся)
system implementation – реалізація системи
execution environment – 1) середовище виконання; 2) елементи процесора
deploy – розгортати, розташовувати
logical grouping – логічна група
extension – розширення
extensively – значно, дуже, широко
relation – *матем.* відношення
metamodel – метамодель, модель високого рівня
since – оскільки
step-by-step – поступовий, ступінчастий, поетапний
business workflow – потік бізнес-операцій
operational workflow – потік функціональних операцій
overall – повний, (у)весь, загальний
flow of control – потік керування
subset – підмножина
sequenced – 1) послідовний; 2) впорядкований
node – вузол
life-span – довговічність

Exercise 39. *Translate the word combinations below into Ukrainian.*

Relationship among the classes; to be split up into components; composite structure diagram; collaboration; deployment diagram;

execution environment; package diagram; step-by-step workflows of components in a system; to show the overall flow of control; state machine diagram; use case diagram; in terms of actors; to be used extensively; a subset of behaviour diagrams; sequenced messages; interaction overview diagram; life-spans of objects; timing diagram.

Exercise 40. *Read and translate text 6.*

Text 6. UML Diagrams

Structure diagrams

Structure diagrams emphasize what things must be in the system being modeled:

Class diagram describes the structure of a system by showing the system's classes, their attributes, and the relationships among the classes.

Component diagram depicts how a software system is split up into components and shows the dependencies among these components.

Composite structure diagram describes the internal structure of a class and the collaborations that this structure makes possible.

Deployment diagram serves to model the hardware used in system implementations, and the execution environments and artifacts deployed on the hardware.

Object diagram shows a complete or partial view of the structure of a modeled system at a specific time.

Package diagram depicts how a system is split up into logical groupings by showing the dependencies among these groupings.

Profile diagram operates at the metamodel level to show stereotypes and profiles. The extension relation indicates what metamodel element a given stereotype is extending.

Since structure diagrams represent the structure they are used extensively in documenting the architecture of software systems.

Behaviour diagrams

Behaviour diagrams emphasize what must happen in the system being modeled:

Activity diagram represents the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

State machine diagram is a standardized notation to describe many systems, from computer programs to business processes.

Use case diagram shows the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases.

Since behaviour diagrams illustrate the behaviour of a system, they are used extensively to describe the functionality of software systems.

Interaction diagrams

Interaction diagrams, a subset of behaviour diagrams, emphasize the flow of control and data among the things in the system being modeled.

Communication diagram shows the interactions between objects or parts in terms of sequenced messages. They represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.

Interaction overview diagram is a type of activity diagram in which the nodes represent interaction diagrams.

Sequence diagram shows how objects communicate with each other in terms of a sequence of messages. Also indicates the life-spans of objects relative to those messages.

Timing diagram is a specific type of interaction diagram, where the focus is on timing constraints.

Exercise 41. *Find in text 6 the English for:*

структурна діаграма; зв'язки між класами; розподілятися на компоненти; діаграма композитних структур; діаграма розгортання; реалізація системи; повна або часткова картина системи; діаграма пакетів; логічна група; профілограма; поетапний потік бізнес-операцій та функціональних операцій компонентів у системі; увесь потік керування; діаграма кінцевого автомату; діаграма прецедентів; підмножина діаграм поведінки; акцентувати увагу на потоці керування і даних між об'єктами у системі, що моделюється; діаграма комунікації; у вигляді послідовності повідомлень; діаграма огляду взаємодії; діаграма послідовностей; довговічність; часова синхронізація; часові обмеження.

Exercise 42. *Match the names of the diagrams with their functions.*

1. Class diagram	a) depicts how a software system is split up into components and shows the dependencies among these components.

2. Object diagram	b) serves to model the hardware used in system implementations, and the execution environments and artifacts deployed on the hardware.
3. Component diagram	c) describes the structure of a system by showing the system's classes, their attributes, and the relationships among the classes.
4. Package diagram	d) shows the functionality provided by a system in terms of actors.
5. Activity diagram	e) shows a complete or partial view of the structure of a modeled system at a specific time.
6. Composite structure diagram	f) depicts how a system is split up into logical groupings by showing the dependencies among these groupings.
7. Deployment diagram	g) represents the business and operational step-by-step workflows of components in a system and the overall flow of control.
8. Use case diagram	h) is standardized notation to describe many systems, from computer programs to business processes.
9. State machine diagram	i) describes the internal structure of a class and the collaborations that this structure makes possible.

Exercise 43. *Use the class diagram of exercise 36 to speak on:*

1. Structure diagrams.
2. Behaviour diagrams.
3. Interaction diagrams.

UNIT 3. COMPUTER HARDWARE

Exercise 1. *Study the basic vocabulary.*

a) terms

terminal device – термінал, термінальний пристрій
storage medium – носій даних
monitor – монітор
motherboard – материнська плата
central processing unit (CPU) – центральний процесор (ЦП)
Random Access Memory (RAM) – оперативна пам'ять, оперативний запам'ятовувальний пристрій (ОЗП)

expansion card (*також* expansion board, PC card) – карта / плата розширення
power supply – блок живлення
Compact Disk Read-Only Memory (CD-ROM) – CD-ROM на компакт-диску
Digital Versatile [Video] Disk Read-Only Memory (DVD-ROM) – DVD-ROM на компакт-диску
CD/DVD-ROM drive – дисковод для компакт(DVD)-дисків
hard disk – жорсткий диск, вінчестер
keyboard – клавіатура
mouse – миша
chipset – мікропроцесорний набір, чіпсет
Basic Input / Output System (BIOS) – базова система введення/виведення
internal bus – внутрішня шина
main memory – оперативна (головна) пам'ять
application – прикладна програма
operating system – операційна система
heat sink – тепловідвідний радіатор *(що його застосовують для запобігання перегріву потужних інтегральних схем)*
fan – вентилятор
boot – початкове завантаження *(комп'ютера)*, самозавантаження
firmware – вмонтоване ПЗ, програмно-апаратні засоби
boot firmware – мікропрограма початкового завантаження
power management – керування електроживленням
power cord – шнур живлення
switch – перемикач
voltage – напруга
disk drive – дисковод
alternating current (AC) – змінний струм
direct current (DC) – постійний струм
socket – розетка, роз'єм, рознім
automation – автоматика

b) nouns
machinery – устаткування *(механічне)*, обладнання
electronics – електроніка, електронні схеми
typewriter – друкарська машинка

brain – мозок, розум
improvement – поліпшення, в/удосконалення
entertainment – розвага, забава
education – виховання, освіта, навчання
field – галузь, сфера діяльності

c) verbs
perform – виконувати
store – запам'ятовувати, зберігати
cover – охоплювати
attach – приєднувати, прикріплювати
enable – робити можливим, уможливлювати, дозволяти
cool – охолоджувати
mediate – бути посередником, бути сполучною ланкою
handle – 1) керувати; 2) мати справу, займатися *(проблемою)*
supply – постачати, подавати
provide – забезпечувати
convert – перетворювати
undergo *(p.* underwent*, pp.* undergone*)* – зазнавати
evolve – розвиватися, еволюціонувати

d) adjectives
appropriate – відповідний; доречний; придатний

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Combine, simple, mouse, convert, opportunity, modern, via, and, network, require, item, manipulate, terminal, personal, operator, occur, requirement.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) expansion card, require, machinery, basic, operation, complex, item, easy, particular, combine, user, firmware, modern;

b) different, advantage, outside, necessary, easily, wide, often, usually, input, progress, modern.

Exercise 4. *Give derivatives of the words below and explain their meanings.*

Model: automate – automation – automatic – automatically

Automate, compute, combine, machine, store, communicate, refer, direct, process, calculate, apply, access, cool, manage, operate, differ, connect, expand, improve, educate, entertain, electron, significant, history, use.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

A functioning computer system; to combine hardware elements with software elements; mechanical devices, machinery and electronics; to perform physical functions; to require three basic hardware items; to perform all data processing; a terminal device, used like a typewriter; two-way communication between the user and the system; a storage medium; to store programs and data; to cover such parts of the personal computer as the monitor, the motherboard, the CPU, the RAM memory, the expansion card, the power supply, the CD-ROM drive, the hard disk, the keyboard and the mouse; to be the "body" of the computer; to be directly attached to the motherboard; to include the central processing unit (CPU), the chipset, the Random Access Memory (RAM), the Basic Input Output System (BIOS) and internal buses; to enable a computer to function; to be cooled by a heat sink and fan; to mediate communication between the CPU and the other components of the system; main memory; boot firmware and power management; to handle tasks; operating system drivers; expansion cards; to convert alternating current to direct current; to provide appropriate voltages to different components; to undergo significant improvements; computation, automation, communication, control, entertainment and education.

Exercise 6. *Read and translate text 1.*

Text 1. Basic Hardware Elements

A functioning computer system combines hardware elements with software elements. The hardware elements are mechanical devices in the system, machinery and electronics that perform physical functions. Usually, a computer system requires three basic hardware items:

1) a personal computer, which performs all data processing;

2) a terminal device, used like a typewriter for two-way communication between the user and the system;

3) a storage medium for storing programs or data.

The term hardware covers such parts of the personal computer as the monitor, the motherboard, the CPU, the RAM memory, the expansion card, the power supply, the CD-ROM drive, the hard disk, the keyboard, and the mouse.

The motherboard is the "body" of the computer. Components are directly attached to the motherboard and include such elements as: the central processing unit (CPU), the chipset, the Random Access Memory (RAM), the Basic Input Output System (BIOS) and internal buses.

The central processing unit (CPU) performs most of the calculations which enable a computer to function, and is sometimes referred to as the "brain" of the computer. It is usually cooled by a heat sink and fan.

The chipset mediates communication between the CPU and the other components of the system, including main memory.

The Random Access Memory (RAM) stores all running processes (applications) and the operating system.

The Basic Input Output System (BIOS) includes boot firmware and power management. The BIOS tasks are handled by the operating system drivers.

Internal buses connect the CPU to various internal components and to expansion cards for graphics and sound.

As for power supply, it includes a power cord, a switch, and a cooling fan and supplies power to the motherboard and internal disk drives. It converts alternating current to direct current and provides appropriate voltages to different components such as the hard disk, the CD-ROM, the motherboard, the CPU socket, etc.

Computer hardware has undergone significant improvements over its history. That is why it has become a platform for uses other than computation, such as automation, communication, control, entertainment, and education. Each field in turn has imposed its own requirements on the hardware, which has evolved in response to the computer uses requirements.

Exercise 7. *Find in text 1 the English for:*

поєднувати елементи апаратного та програмного забезпечення; механічні пристрої, механічне обладнання та електроніка; потребувати трьох основних апаратних елементів; виконувати оброблення даних; двосторонній зв'язок між користувачем та системою; носій

даних; зберігати програми та дані; карта розширення, материнська плата та оперативна пам'ять; джерело живлення та дисковод для компакт-дисків; жорсткий диск, клавіатура та миша; центральний процесор, мікропроцесорний набір та базова система введення/виведення; охолоджуватися тепловідвідним радіатором та вентилятором; зберігати прикладні програми та операційну систему; включати мікропрограму початкового завантаження та керування електроживленням; перетворювати змінний струм на постійний; забезпечувати потрібну напругу.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. The software elements are mechanical devices in the system, machinery and electronics that perform physical functions. 2. As a rule, computer system needs two basic hardware items to work perfectly. 3. The term hardware covers such parts of the personal computer as the monitor, the motherboard, the CPU, the RAM memory, the expansion card, the power supply, the CD-ROM drive, the hard disk, the keyboard, and the mouse. 4. Hardware elements are directly attached to the monitor. 5. The Basic Input Output System tasks are handled by the operating system drivers. 6. Internal buses connect the CPU to various internal components and to expansion cards for graphics and sound. 7. Computer hardware has undergone significant improvements over its history.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to combine	a) two-way communication
2) to require	b) the motherboard
3) to be used for	c) significant improvements
4) to be directly attached to	d) three basic hardware items
5) to be handled by	e) hardware elements with software elements
6) to include	f) calculations
7) to undergo	g) the operating system drivers
8) to perform	h) "the brain" of the computer
9) to be referred to as	i) alternating current to direct current
10) to convert	j) boot firmware and power management

Exercise 10. *Fill in the blanks with prepositions* ***on, of, in, to, with, over, by*** *where necessary.*

1. A functioning computer system combines hardware elements … software elements. 2. The hardware elements are mechanical devices … the system, machinery and electronics that perform … physical functions. 3. Power supply supplies power … the motherboard and internal disk drives. 4. The motherboard is the "body" … the computer. 5. Components are directly attached … the motherboard. 6. The BIOS tasks are handled … operating system drivers. 7. Computer hardware has undergone significant improvements … its history. 8. Each field has imposed its requirements … the hardware. 9. Components are attached … the motherboard.

Exercise 11. *Fill in the blanks with proper terms (****the hardware elements, a power supply, the central processing unit, the BIOS, the motherboard, a personal computer, a terminal device****) to complete the sentences.*

1. _____are the mechanical devices in the system, the machinery and the electronics that perform physical functions. 2._____ performs all data processing. 3. _____ is used like a typewriter for two-way communication between the user and the system. 4. _____ is the "body" of the computer. 5. _____ performs most of the calculations which enable a computer to function. 6. _____ includes boot firmware and power management. 7. _____ includes power cord, switch, and cooling fan and supplies power at appropriate voltages to the motherboard and internal disk drives.

Exercise 12. *Answer the questions on text 1.*

1. What combines different elements of the PC? 2. What are the hardware elements? 3. How many basic hardware items does a computer system usually require? 4. What does the term hardware cover? 5. What is the "body" of the PC? 6. What components are directly attached to the motherboard? 7. What function does the central processing unit perform? 8. What is the "brain" of the PC? 9. What is the CPU usually cooled by? 10. What kind of communication does a chipset mediate? 11. What does the RAM store? 12. What does the BIOS include? 13. What are the BIOS tasks handled by? 14. What connects the CPU to various internal components? 15. What does the power supply include?

16. What is the power supply designed for? 17. Why has computer hardware become a platform for different uses? 18. What kinds of uses has hardware become a platform for?

Exercise 13. *Make up questions to the italicized parts of the sentences.*

1. ***Computer hardware*** has undergone significant improvements over its history. 2. Computer hardware has become ***a platform*** for ***different uses.*** 3. Usually, ***a computer system*** requires ***three*** basic hardware items. 4. Components are ***directly*** attached to ***the motherboard***. 5. ***The CPU*** is usually cooled by ***a heat sink and fan***. 6. ***The hardware*** elements perform physical functions. 7. The CPU performs ***most of the calculations.*** 8. ***The CPU*** is sometimes referred to as ***the "brain" of the computer.***

Exercise 14. *Translate into English.*

1. Функціональна комп'ютерна система поєднує елементи апаратного та програмного забезпечення. 2. Як правило, комп'ютерна система потребує трьох основних елементів: персонального комп'ютеру, терміналу та носія даних. 3. Персональний комп'ютер виконує оброблення даних. 4. Термінал використовують як друкарську машинку для двостороннього зв'язку між користувачем та системою. 5. Носій даних використовують для зберігання програм і даних. 6. Термін «апаратне забезпечення» охоплює такі елементи персонального комп'ютера, як: материнська плата, монітор, центральний процесор, оперативна пам'ять, карта розширення, жорсткий диск, блок живлення, дисковод для компакт-дисків, клавіатура та комп'ютерна миша. 7. Материнська плата є головною частиною комп'ютера. 8. Компоненти, що кріпляться безпосередньо до материнської плати, включають центральний процесор, мікропроцесорний набір, оперативну пам'ять, базову систему введення/виведення та внутрішні шини. 9. Центральний процесор виконує більшу частину обчислень, які дозволяють комп'ютеру функціонувати. 10. Центральний процесор зазвичай охолоджується з допомогою теплопровідного радіатора та вентилятора. 11. Мікропроцесорний набір є сполучною ланкою між центральним процесором та іншими компонентами системи, включаючи оперативну пам'ять. 12. В оперативній пам'яті зберігаються всі прикладні програми та операційна система. 13. Базова система введення/виведення вклю-

чає мікропрограму початкового завантаження та керування електроживленням. 14. Завданнями базової системи введення/виведення займаються драйвери операційної системи. 15. Внутрішні шини з'єднують центральний процесор з різними внутрішніми компонентами та картами розширення для графічного та звукового відображення. 16. Блок живлення включає шнур живлення, перемикач та охолоджувальний вентилятор. 17. Блок живлення забезпечує електроживлення материнської плати та внутрішніх дисководів. 18. Він перетворює змінний струм на постійний, а також подає потрібну напругу до таких компонентів, як: жорсткий диск, CD-ROM, материнська плата, роз'єм центрального процесора. 19. Апаратне забезпечення комп'ютера за свою історію зазнало значних удосконалень. 20. Комп'ютер є основою для виконання не лише обчислень, але й автоматизації, зв'язку, керування, засобом розваг та освіти.

Exercise 15. *Retell the text "Computer Hardware".*

Exercise 16. *Study the ways of constructing Indefinite, Perfect and Continuous Passive verb forms.*

You use the Passive Voice when you want to focus on the person or thing affected by the action, rather than on the "doer" of the action (or agent).

*Our house **was built** a hundred years ago.*
*My notebook **has been stolen.***

We use appropriate form of the verb ***to be*** **+** ***Past Participle*** of the main verb.

Present Indefinite	More and more cars **are sold** every year.
Past Indefinite	The Internet **was developed** in the 1960s.
Future Indefinite	Everyone who applies **will be given** an interview.
Present Continuous	The conference **is being held** next weekend.
Past Continuous	Every car which left the ferry **was being stopped and searched.**
Present Perfect	We **have been invited** to Paul's presentation.
Past Perfect	He **had been promoted** three times before becoming a director.
Future Perfect	The article **will have been written** by two o'clock tomorrow.

Be going + Passive Infinitive	The new company **is going to be opened** by the President.
Modal verbs + Passive Infinitive	Books **must be returned** within three weeks.
Modal verbs + Perfect Passive Infinitive	The office **may have been left** open deliberately.

Exercise 17. *Change the following sentences into the Passive Voice*.

Model:	We **bought** this netbook two years ago. – This netbook **was bought** two years ago by us.

1. Smart cards store vital information such as health records, drivers' licenses, bank balances, and so on. 2. Browsers may search, view, and even add and edit data on the World Wide Web. 3. The firm hired an enterprise architect to oversee the development of the new software platform. 4. A malicious program will crash and terminate software with a confusing message. 5. Almost all computers, including hand-held computers, desktop computers, supercomputers, and even video game consoles, use an operating system. 6. The university used encryption on its web site to protect information from unauthorized access. 7. In the future, digital distribution on the Internet will replace all other forms of media distribution including CDs, DVDs, and even radio and television broadcasts. 8. Users of the Internet can send mail messages with vast databases of information to each other. 9. Mobile devices normally provide the programs needed to enable Wi-Fi, Bluetooth, or other wireless connectivity. 10. Years ago, people commonly set up their home networks just to connect a few PCs, share some documents and perhaps a printer. 11. A scanner can take a photograph or magazine article and digitize it. 12. The central processing unit (CPU) performs most of the calculations which enable a computer to function. 13. Windows 8 is now available, but most organizations are still deploying Windows 7.

Exercise 18. *Change the following sentences into the Passive Voice* and *ask questions on them.*

1. Small electronic device can control every move of your robotic personal assistant. 2. In the 2000s smartphones, cloud computing, and other innovations revolutionized the way we live and work. 3. In the near future, computers will use nanotechnology to shrink the size of silicon

chips, increasing speed and power with parallel processing. 4. The military is currently working on the next generation of future computers. 5. The system analyst was analyzing and interpreting company's needs at that time. 6. Practical computer systems divide software systems into three major classes: system software, programming software, and application software. 7. Cellular phones are now also dialing up the Internet to provide e-mail and answering machine services. 8. School innovators will require some patience as computer skills develop. 9. Our new programmer was developing software for the operating system of a computer yesterday evening. 10. The ARPANET computer network made a large contribution to the evolution of e-mail. 11. The Operating System manages and controls the computer through the use of an interface. 12. Past architecture research often focused on chip microprocessors or stand-alone computers with performance as main optimization goal. 13. Information and communication technology (ICT) is transforming our world, including healthcare, education, science, commerce, government, defense, and entertainment. 14. In the future, PDA will store the entire human knowledge base. 15. Various computer professionals were writing programs in the computer department.

Exercise 19. Change the following sentences into the Active Voice. Add the doer/agent of the action where necessary.

1. Bill Gates was known in computer circles as a founder of Microsoft Corporation and developer of Windows. 2. Mathematical models can be used not only in engineering disciplines but also in the social sciences. 3. Solaris was developed by Sun Microsystems as a more open option of SunOS for its SPARC-based servers and workstations. 4. For hand-held and desktop computers, the user interface is generally considered a part of the operating system. 5. Software programs are normally written and compiled for certain hardware platforms. 6. Mainframes will be completely replaced by the lesser categories of mid-range computers, workstations, and powerful PCs. 7. Multimedia systems are known for their educational and entertainment value. 8. Modern technologies are being researched and developed including biocomputers and quantum computers. 9. Networks with speeds of 100 gigabits and higher are being planned using fiber optics for the Internet 2 system. 10. An alliance between IBM, Motorola, and Apple was formed several years ago to develop and manufacture the PowerPC chip. 11. Microcomputers can be

configured to serve multiple users. 12. Today thousands of applications are designed for almost every purpose, from writing letters to playing games. 13. Mobile computing can be employed when we interact with our smartphones. 14. Future Internet appliances will be based on embedded computing systems. 15. The computer system was being upgraded by the administrator all morning yesterday.

Exercise 20. *Define the tense and voice used in the following sentences (Present, Past, Future Indefinite / Continuous Active/Passive) and translate them.*

1. Programmers are also known as 'software developers'. 2. The user was advised to reboot the computer after a serious crash in which the computer no longer responded. 3. Embedded systems are typically controlled by inexpensive, specialized processors which can only handle very specific tasks. 4. More and more, software is being distributed over the Internet, as open source, shareware, freeware, or traditional proprietary and upgrade versions. 5. By the early 1990's, computers were commonly used for writing papers, playing games, financial accounting, and business productivity applications. 6. Each Windows OS is optimized for different users, hardware configurations, and tasks. 7. Since main memory is volatile, all contents are lost when the computer power is turned off. 8. In what way will computers be linked on a network? 9. The notebook was performing slowly when changing programs, so the technician installed more RAM and this solved the problem. 10. The original backbone of the Internet is based on an old military network called ARPANET which was built by ARPA in the late 1960's. 11. Technology will offer some great capabilities in pulling together the design of any software system. 12. Not surprisingly, Linux's OS market share is growing very quickly around the world. 13. What operating system are you installing? – Windows Vista. 14. Since nobody likes to wait for a computer, high-quality computers will have fast processors and lots of quick memory. 15. The letter is being sent now.

Exercise 21. *Complete these sentences using the appropriate passive form of the verbs in brackets (Present, Past or Future Indefinite / Continuous).*

1. Applications (mean) to make users more productive and get work done faster. 2. An embedded OS can (find) inside an increasing number

of consumer gadgets including phones (iPhone OS), PDAs (Windows CE), and digital media players. 3. The way that the motherboard (design) and (lay out) dictates how the entire computer is going (organize). 4. The instructions (perform) by a computer while the programs (debug) and (test). 5. Intel (start) by several people who are now legends in the computer world, including Robert Noyce and Gordon Moore. 6. Bluetooth technology (optimize) for networking between common consumer electronics such as mobile phones, mp3 players, and similar devices. 7. We project that mainframe computer applications for businesses essentially (replace), either by client/server applications or new language software. 8. The antivirus programs currently (install) by a specialist. 9. The Macintosh operating systems (design) to (use) on Apple Macintosh computers. 10. The computer contacted the address and the information (make) available to the operator. 11. Most users have no idea that their information (collect) and (store) at the moment on their computer. 12. Ultimately people's power (must exercise) to ensure that computers (use) not only efficiently but in a socially responsible way. 13. Regardless of the networking choice, future Internet appliances (base) on embedded computing systems. 14. When nanotechnology fully (develop), machines (assemble and snap) together using the chemistry of atoms and molecules. 15. New applications program (write) by the program developer yesterday evening.

Exercise 22. *Use Indefinite or Continuous Tenses (Present, Past or Future) of the verbs in brackets choosing between the Active and Passive forms.*

1. A company or organization (store) its information in electronic documents on one of the Internet computers. 2. The whole exciting Internet world (wait) for you! 3. In some families children (not allow) to surf the net on their own. 4. When the word processor application (crash), the user (have) to abort the program and (lose) all his unsaved changes. 5. As a student of English and Technology, you (hear) people use the words 'Internet' and 'World Wide Web' almost interchangeably. 6. In the future, all forms of media distribution including CDs, DVDs, and even radio and television broadcasts (replace) by digital distribution on the Internet. 7. Digital cameras (become) more common. 8. The program (design) by the software engineer now. 9. When the network administrator (format) the wrong hard disk drive array, he accidentally (lose) all the company data. 10. The man (not can) spell-check docu-

ments in German last Saturday because his copy of Word (upgrade). 11. The computer user (not can) access a computer resource because he (forget) his username and password. 12. You (find out) the answer to almost everything on Google nowadays. 13. When you (surf) the net, you (move) from one document or the web site to another. 14. Computer science experts predict that more traditional desktop computer (replace) by powerful and affordable laptop computers and netbooks. 15. The first version of a software application (not release) to the public because it (contain) serious bugs. 16. In recent years, more and more features (include) in the basic GUI OS, including notepads, sound recorders, and even web browsers and games. 17. Sixty years ago computers also (cost) a lot of money to build and operate and they only (use) by large organizations. 18. Computers all over the world (join) by phone lines, satellite or cable. 19. Students (download) several files from the university's server yesterday morning. 20. This time next week our department (work) on this project. 21. When computers first (appear), they (take up) whole rooms and (require) specialized training to operate them. 22. The hackers (admit) deleting new programs while they (interview) by security police. 23. PC (protect) from different viruses by special programs. 24. The commercial global Internet system with its World Wide Web applications (begin) in 1993-1994. 25. This time tomorrow the program designer (write) specifications for new computer systems.

Exercise 23. *Read and translate text 2.*

Text 2. Hardware Categories

A typical computer consists of two parts: hardware and software. Hardware is any electronic or mechanical part of the computer system that you can see or touch. It is categorized as follows:

Input hardware is used to collect data and input it into the computer system in computer-usable form. The keyboard and mouse are the most common input devices.

Processing hardware retrieves and executes (interprets) instructions (software) provided to the computer. The main components of processing hardware are the central processing unit (CPU), which is the brain of the computer, and main memory, where all instructions and/or data ready for processing are held. Since main memory is volatile, all contents are lost when the computer's power is turned off.

Output hardware provides a means for the user to view information produced by the computer system – either in hardcopy form, such as printouts from a printer, or softcopy form, such as a display on a monitor, a TV-like screen.

Communications hardware facilitates connections between computers and computer systems over phone lines and other channels. Examples are modems, cables and fax modems.

Software is a set of instructions, called a program, which tells a computer what to do. Software that runs the hardware and allows the computer to manage its resources is system software. Software that performs a general business function is referred to as applications software.

Exercise 24. *Fill in the blanks with proper terms from text 2 to complete the sentences.*

1._____ refers to any electronic or mechanical part of the computer system that you can see or touch. 2._____ a set of instructions, called a program, which tells a computer what to do. 3._____ runs the hardware and allows the computer to manage its resources. 4._____ performs a general business function. 5._____ retrieves and executes instructions provided to the computer. 6._____ facilitates connections between computers and computer systems over phone lines and other channels. 7._____ is used to collect data and input it into the computer system in computer-usable form. 8._____ provides a means for the user to view information produced by the computer system.

Exercise 25. *Speak on computer hardware adding some information you know from your own experience.*

Exercise 26. *Read, translate and entitle text 3.*

Text 3

Nowadays PCs represent the wide-spread class of digital machines. There are many different individual components which can be mixed and matched in thousands of different configurations. This lets you customize the PC you either buy or build to meet your exact needs.

The system case, sometimes called the chassis or enclosure, is the metal and plastic box that houses the main components of the computer. Most people do not consider it a very important part of the computer (perhaps in the same way they would not consider their own skin a very important body organ). While the case is not as critical to the system as some other computer components (like the processor or hard disk), it has several important roles to play in the functioning of a properly-designed and well-built computer.

The case does not appear to perform any function at all, at first glance. However, this definitely is not true; the case is in fact much more than just a box. The motherboard is mounted into the case, and all the other internal components are mounted into either the motherboard or the case itself. The case must provide a solid structural framework for these components to ensure that everything fits together and works well.

The system case performs several important functions for your PC, for instance, protection for the computer circuits, cooling and system organization.

Exercise 27. *Put key questions to text 3.*

Exercise 28. *Read and translate text 4.*

Text 4. Motherboard

The motherboard is, in many ways, the most important component in your computer (not the processor, even though the processor gets much more attention.). The motherboard and its major components (the chipset, BIOS, cache*, etc.) are the major systems that this brain uses to control the rest of the computer. Having a good understanding of how the motherboard and its contained subsystems work is probably the most critical part of getting a good understanding of how PCs work in general.

The motherboard plays an important role in all aspects of the computer system. In one way or another, everything is eventually connected to the motherboard. The way that the motherboard is designed and laid out dictates how the entire computer is going to be organized.

The motherboard contains the chipset and BIOS program, which control most of the data flow within the computer. Almost all communication between the PC and its peripherals, other PCs, and the user goes

* cache – надоперативна пам'ять, кеш, надоперативний ЗП

through the motherboard. The motherboard dictates directly the choice of processors, memory, system buses for use in the system, and these components affect the system's performance. It also determines what types of peripherals you can use in your PC.

The capabilities of the motherboard specify to what extent you will be able to upgrade your machine. For example, there are some motherboards that will accept regular Pentiums of up to 133 MHz speed only, while others will go to 200 MHz. Obviously, the second one will give you more room to upgrade when starting with a P133.

Exercise 29. *Write a summary of the text "Motherboard".*

Exercise 30. *Say whether the statements below are true or false. Correct the false ones.*

1. Nowadays PC represents the useful part of design idea of every living-room. 2. The system case is usually one of the most overlooked parts of the PC. 3. The main functions of the system case are to protect the computer circuits, cooling system and repair the motherboard. 4. Any computer system can't work without the motherboard as it is its heart. 5. The chipset and other motherboard circuitry are the liver of the motherboard. 6. The main function of the chipset is to direct traffic and control the flow of information inside the computer. 7. The system buses are the electrical channels through which various parts of the computer communicate. 8. The BIOS is a computer program that helps you to learn hardware. 9. The BIOS gives you the opportunity to set or change many different parameters that control how your computer will function. 10. The system cache is not large, and we can find it between the processor and the system memory. 11. Each time the processor requests great amounts of meals. 12. System case, motherboard and system devices are personal computer components.

Exercise 31. *Agree or disagree with the following statements. Give your reasons.*

1. Using a computer helps us to develop our language skills.
2. Computers have greatly changed the way we work and study.

Exercise 32. *Make a list of advantages and disadvantages of using a personal computer. Discuss it with your group-mates.*

Exercise 33. *Find some additional information and speak on the topics:*

1. The CPU.
2. The main memory (RAM and ROM).
3. Peripherals (input devices, output devices, storage devices).

UNIT 4. OPERATING SYSTEMS

Exercise 1. *Study the basic vocabulary.*

a) terms

share (*resources*) – розподіляти, спільно використовувати (*ресурси*)
host – головна система, хост-система
master system – головна, центральна система
hand-held computer – кишеньковий комп'ютер
video game console – приставка для відеоігор
embedded (built-in) operating system – вмонтована / вбудована операційна система
video/visual display unit (VDU) – монітор, дисплей
real-time operating system – операційна система реального часу
single-user operating system – операційна система індивідуального користування
single-task(ing) operating system – однозадачна операційна система
multi-tasking operating system – багатозадачна операційна система
multi-user operating system – багатокористувацька операційна система
multitasking – багатозадачність
graphical user interface – графічний інтерфейс користувача
word processing – оброблення текстів
spreadsheet – великоформатна (*електронна*) таблиця
move data – пересилати дані / інформацію
networking – 1) робота / спілкування в мережі; мережевий режим
hard drive, hard disk – жорсткий диск, вінчестер
universal serial bus (USB) – універсальна послідовна шина
USB key, USB drive, flash drive – флеш-накопичувач / флеш-пам'ять
file system – файлова система
scalable processor architecture (SPARC) – архітектура процесора, яку можна нарощувати
dumb terminal – простий (*неінтелектуальний*, «*німий*») термінал

b) nouns
instrument – прилад
purpose – мета, намір
majority – більшість
market share – питома вага даного товару на ринку
option – варіант, версія, опція

c) verbs
act as smth – виконувати функцію чогось
run – виконувати
relieve – позбавляти, звільняти
load – завантажувати
communicate – передавати
accept – приймати
design – призначати
offer – пропонувати
intend – призначати, планувати, мати намір
dedicate – призначати
split – ділити на частини
switch (*between*) – перемикати
enable – давати можливість, уможливлювати, дозволяти
serve to – подавати

d) adjectives
responsible – відповідальний
master – головний, провідний
broad – широкий
recent – останній, найновіший
powerful – потужний
expensive – дорогий

e) adverbs
simultaneously – одночасно
currently – тепер, зараз, нині

f) prepositions
unlike – на відміну (*від чогось*)
within – 1) в, в межах; 2) в/усередині

g) conjunctions
whether – чи
however – однак, проте

Exercise 2. *Choose verbs among the following words. Put the first letters of the verbs into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Operating, manipulate, priority, include, contain, multitasking, system, receive, expensive, offer, software, simplify, responsible, operate, hand-held, find, multi-user, tend.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) manage, make easier, write programs, hand-held computer, however, execute, display, communicate the message, accept, machinery, single-user, design, laptop computer, enable, common, split, locate, develop, host, easy, apply, operate, affect, significant, built-in, memory;

b) easy, significant, input, responsible, centre, good, single-user, all, include, oldest, embedded, effectively, different.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: simple – simplify – simplification – simplicity – simply

Simple, ease, manage, coordinate, apply, operate, inform, execute, science, effect, store, inform, process, signify, differ, major.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

To be responsible for; the management, coordination and sharing of computer resources; to act as a host for applications that are run on the machine; a master system; to manage the basic operation of the computer; to relieve applications from having to control the hardware; to make it easier to write programs; hand-held computers and video game consoles; embedded operating system; some features; to execute a program; to print or display the result of a program on the printer or the screen; store the output data or programs; to communicate the message from the system to the user; to accept input from the user through the

keyboard or mouse; to be designed to manage the computer; single-user, multi-tasking operating system; to take advantage of the computer resources simultaneously; the most common operating systems; built-in networking support; multi-user environment; regardless of the media; SPARC-based servers; a powerful and expensive computer; to be built for serving information to many PCs or dumb terminals; to have the largest share of the Internet market.

Exercise 6. *Read and translate text 1.*

Text 1. Operating System Functions and Categories

An operating system is an interface between the hardware and the user. It is responsible for the management, coordination and sharing of computer resources. The operating system acts as a host for applications that are run on the machine. It is the master system of programs that manages the basic operation of the computer. This relieves applications from having to control the hardware and makes it easier to write programs. Almost all computers, including hand-held computers, desktop computers, supercomputers, and even video game consoles, use an operating system. Some of the oldest models may however use an embedded operating system on a compact disk or other data storage devices.

A good operating system should have the following features:

1) help in loading of programs and data from external sources into the internal memory before they are executed; 2) help programs to perform input/output operations, such as: a) print or display the result of a program on the printer or the screen; b) store the output data or programs written on the computer in a storage device; c) communicate the message from the system to the user through the VDU; and d) accept input from the user through the keyboard or mouse.

The broad categories of operating systems are: 1) real-time operating systems used to control machinery, scientific instruments and industrial systems; 2) single-user, single-task operating systems designed to manage the computer so that one user can effectively do one thing at a time; 3) single-user, multi-tasking operating systems used by most people on their desktop and laptop computers today. Windows and Mac OS are examples of an operating system that will let a single user have several programs in operation at the same time, and 4) multi-user operating systems allow many different users to take advantage of the computer resources simultaneously.

The most common operating systems include Microsoft Windows, Mac OS, UNIX, Linux and Solaris. Windows lets you display your work in windows. A window is a portion of the video display area dedicated to some specific purpose. With Windows, which supports multitasking and graphical user interface, you can display several windows on a computer screen, each showing a different application, such as word processing or spreadsheet. You can easily switch between the applications and move data between them. Windows 7 is the recent version of Microsoft Windows family. Microsoft Windows has a significant majority of market share in the desktop and notebook computer markets, while servers generally use Linux or other Unix-like systems. Embedded device markets are split among several operating systems.

The Macintosh operating system was designed to be used on Apple Macintosh computers. This operating system supports multi-tasking and enables users to read MS-DOS and Windows files. The UNIX operating system is one of the oldest operating systems. It is a multi-tasking operating system that includes built-in networking support. It is a popular operating system in universities, where a multi-user environment is often needed. Unlike other operating systems, Linux and UNIX allow any file system to be used regardless of the media it is stored in, whether it is a hard drive, a disc (CD, DVD...), a USB key, or even contained within a file located on another file system.

Solaris was developed by Sun Microsystems as a more open option of SunOS for its SPARC-based servers and workstations. Sun machines are popular, powerful, and expensive computers built for serving information to many PCs or dumb terminals. Their processors can perform several tasks simultaneously. Many universities and large corporations use Sun machines to serve information on their networks. Currently, Solaris is one of the most popular versions of UNIX and has the largest share of the Internet market.

Exercise 7. *Find in text 1 the English for:*

інтерфейс між апаратним забезпеченням та користувачем; відповідати за керування, координацію та розподіл ресурсів; головна система; звільняти прикладні програми від необхідності керувати апаратними засобами; полегшувати написання програм; кишенькові комп'ютери та приставки для відеоігор; вмонтована (вбудована) операційна система; пристрої зберігання даних; мати такі власти-

вості; операційна система реального часу; прилади для наукових досліджень та промислові системи; операційна система індивідуального користування; передавати повідомлення від системи до користувача; приймати вхідну інформацію; багатозадачна операційна система; багатокористувацька система; дозволяти різним користувачам одночасно використовувати комп'ютерні ресурси; найпоширеніші операційні системи; частина площі відеодисплея, що має певне призначення; підтримувати багатозадачність та графічний інтерфейс користувача; оброблення текстів та великоформатна (електронна) таблиця; перемикати прикладні програми та пересилати інформацію між ними; остання версія; домінувати на ринку настільних комп'ютерів та ноутбуків; вмонтований пристрій; вмонтована підтримка мережевого режиму; багатокористувацьке середовище; використовуватися незалежно від носія; потужні та дорогі комп'ютери; подавати інформацію багатьом ПК; одночасно виконувати кілька завдань.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. An operating system is an interface between the hardware and the user. 2. It is responsible for the management, coordination and sharing of computer resources. 3. It is the master system of programs that manages the basic operation of the software. 4. The operating system acts as a host for system programs that are run on the machine. 5. The operating system relieves applications from having to control the hardware but makes it more difficult to write programs. 6. Almost all computers, including hand-held computers, desktop computers, supercomputers, and even video game consoles, use an operating system. 7. A good operating system should help in loading of programs and data from external sources into the internal memory before they are executed and help programs to perform input/output operations. 8. Windows and Mac OS are multi-user operating systems that allow many different users to take advantage of the computer resources simultaneously. 9. The most common operating systems include Mac OS, UNIX, Linux and Solaris. 10. Windows lets you display your work in cells. 11. A window is a portion of the video display area dedicated to some specific purpose. 12. With Windows, which supports multitasking and graphical user interface, you can display several windows on a computer screen, each showing a dif-

ferent application, such as word processing or spreadsheet. 13. Windows 2006 is the recent version of Microsoft Windows family. 14. Microsoft Windows has a significant majority of market share in the servers and notebook computer markets. 15. The Macintosh operating system was designed to be used on Apple Macintosh computers. 16. The Macintosh operating system does not support multi-tasking and enables users to read Windows files. 17. The UNIX operating system is one of the oldest operating systems. 18. UNIX is a single-user operating system that includes built-in networking support. 19. Linux and UNIX allow any file system to be used regardless of the media it is stored in. 20. Solaris was developed by Sun Microsystems as a more open option of SunOS for its SPARC-based servers and workstations. 21. Sun machines are popular, powerful, and expensive computers built for serving information to many mainframes and supercomputers. 22. Sun machines processors can perform several tasks simultaneously. 23. Currently, Solaris is one of the most popular versions of UNIX and has the largest share of the Internet market.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to manage	a) computer resources
2) to control	b) several operating systems
3) to store	c) multitasking
4) to take advantage of	d) basic operation of the computer
5) to switch between	e) serving information to PCs
6) to display	f) the output data or programs
7) to support	g) machinery and scientific instruments
8) to be split among	h) the work in windows
9) to be built for	i) several tasks simultaneously
10) to perform	j) the applications

Exercise 10. *Fill in the blanks with prepositions* ***to, in, into, on, of, for, from, between, among, through, at, by*** *where necessary.*

1. An operating system is an interface … the hardware and the user. 2. It is responsible … the management, coordination and sharing … computer resources. 3. The operating system acts as a host … applications that are

run … the machine. 4. It is the master system … programs that manage the basic operation … the computer. 5. This relieves applications … having to control the hardware.. 6. A good operating system should help … loading … programs and data … external sources … the internal memory. 7. Operating system should communicate the message … the system … the user … the VDU and accept input … the user … the keyboard or mouse. 8. Single-user, multi-tasking operating systems are used … most people … their desktop and laptop computers today. 9. Windows and Mac OS are examples … an operating system that will let a single user have several programs … operation … the same time. 10. A window is a portion … the video display area dedicated … some specific purpose. 11. You can easily switch …the applications and move data … them. 12. Microsoft Windows has a significant majority … market share … the desktop and notebook computer markets. 13. Embedded device markets are split … several operating systems. 14. Unlike other operating systems, Linux and UNIX allow any file system to be used regardless … the media it is stored … . 15. Solaris was developed … Sun Microsystems as a more open option … SunOS … its SPARC-based servers and workstations. 16. Sun machines are popular, powerful, and expensive computers built … serving information … many PCs or dumb terminals.

Exercise 11. *Fill in the blanks with proper terms* ***(operating system, Unix, Windows, multi-user operating system, Macintosh operating system, Sun machines, window, Solaris)*** *to complete the sentences.*

1. _____ is a portion of the video display area dedicated to some specific purpose. 2. _____ is an operating system that allows many different users to take advantage of the computer resources simultaneously. 3. _____ are popular, powerful, and expensive computers built for serving information to many PCs or dumb terminals. 4. _____ is one of the most popular versions of UNIX developed by Sun Microsystems as a more open option of SunOS. 5. _____ is one of the oldest multi-tasking operating systems that includes built-in networking support. 6._____ is an operating system that lets you display your work in windows. 7. _____ is an interface between the hardware and the user. 8. _____ is an operating system that was designed to be used on Apple-Macintosh computers.

Exercise 12. *Answer the questions on text 1.*

1. What is an operating system? 2. What is it responsible for? 3. What are the functions of the operating system? 4. What does the operating system relieve applications from? 5. What kinds of computers use an operating system? 6. What kind of operating system do the old models use? 7. What features should a good operating system have? 8. What are the broad categories of operating systems? 9. What are real-time operating systems used for? 10. What are single-user, single-task operating systems designed for? 11. What operating systems are used by most people on their desktop and laptop computers today? 12. What operating systems allow many different users to take advantage of the computer resources simultaneously? 13. What are the most common operating systems? 14. What does Windows allow the users to do? 15. What operating system has a significant majority of market share in the desktop and notebook computer markets? 16. What operating systems do servers generally use? 17. What files does the Macintosh operating system enable users to read? 18. What kind of system is the UNIX operating system? 19. How do Linux and UNIX differ from other operating systems? 20. What kind of computers was Solaris developed for? 21. What kind of computers are Sun machines? 22. What do many universities and large corporations use Sun machines for? 23. What operating system has the largest share of the Internet market?

Exercise 13. *Put all possible questions to the sentences below.*

1. An operating system is an interface between the hardware and the user. 2. An operating system is responsible for the management, coordination and sharing of computer resources. 3. The operating system relieves applications from having to control the hardware. 4. Almost all computers, including hand-held computers, desktop computers, supercomputers, and even video game consoles, use an operating system. 5. Some of the oldest models may however use an embedded operating system. 6. A good operating system should help in loading of programs and data from external sources into the internal memory before they are executed. 7. The most common operating systems include Microsoft Windows, Mac OS, UNIX, Linux and Solaris. 8. You can display several windows on a computer screen. 9. The Macintosh operating system was designed to be used on Apple Macintosh computers. 10. This oper-

ating system supports multi-tasking and enables users to read MS-DOS and Windows files. 11. Their processors can perform several tasks simultaneously.

Exercise 14. *Translate into English.*

1. Операційна система – це інтерфейс між апаратним забезпеченням та користувачем. 2. Операційна система відповідає за керування комп'ютерними ресурсами, їх координацію та розподіл. 3. Це головна система серед програм, що керують роботою комп'ютера. 4. Операційна система звільняє прикладне програмне забезпечення від необхідності керувати апаратним забезпеченням та полегшує написання програм. 5. Операційні системи реального часу використовують для керування механічним обладнанням, приладами для наукових досліджень та промисловими системами. 6. Більшість людей використовують у своїх настільних комп'ютерах та ноутбуках багатозадачні операційні системи індивідуального користування. 7. Існують такі класи операційних систем: операційні системи реального часу; однозадачні операційні системи індивідуального користування; багатозадачні операційні системи індивідуального користування та багатокористувацькі операційні системи. 8. Найпоширеніші операційні системи включають Microsoft Windows, Mac OS, UNIX, Linux та Solaris. 9. «Вікно» – це частина площі відеодисплея, призначена для певної мети. 10. Windows підтримує багатозадачність та графічний інтерфейс користувача. 11. З Windows можна переходити від однієї програми до іншої та пересилати інформацію між ними. 12. Microsoft Windows переважає на ринку настільних комп'ютерів та ноутбуків, тоді як сервери зазвичай використовують Linux чи інші системи, подібні до UNIX. 13. На відміну від інших операційних систем, Linux та UNIX дозволяють використовувати будь-яку файлову систему, незалежно від носія, на якому вона зберігається. 14. Solaris була створена компанією Sun Microsystems як більш відкрита версія SunOS для її серверів та робочих станцій. 15. Машини Sun є популярними, потужними машинами, створеними для надання інформації багатьом ПК та простим терміналам. 16. На сьогодні Solaris є однією з найбільш популярних версій UNIX і домінує на Інтернет ринку.

Exercise 15. *Retell the text "Operating Systems".*

Exercise 16. *Study the vocabulary to text 2.*

Application programming interface (API) – інтерфейс прикладного програмування
system call – системний виклик
nvoke – викликати, запускати, активізувати програму
user interface (UI) – користувацький інтерфейс, інтерфейс користувача
command line interface (CLI) – інтерфейс командного рядка, командний інтерфейс
kernel – ядро *(частина операційної системи, що виконує найбільш важливі завдання)*
request – робити запит
assign – призначати, розподіляти
establish – встановлювати
priority – пріоритет, порядок черговості
port – переносити (*напр., програму з однієї машини на іншу*)

Exercise 17. *Read and translate text 2.*

Text 2. Operating System Interfaces

Operating systems offer a number of services to application programs and users. Applications access these services through application programming interfaces (APIs) or system calls. By invoking these interfaces, the application can request a service from the operating system, pass parameters, and receive the results of the operation. Users may also interact with the operating system with some kind of software user interface (UI) like typing commands by using command line interface (CLI) or using a graphical user interface (GUI, commonly pronounced “gooey”). For handheld and desktop computers, the user interface is generally considered a part of the operating system. On large multi-user systems like UNIX and UNIX-like systems, the user interface is implemented as an application program that runs outside the operating system.

The operating system acts as an interface between an application and the hardware. The user interacts with the hardware from "the other side". The operating system is a set of services which simplifies the development of applications. Executing a program involves the creation of a process by the operating system. The kernel creates a process by assigning memory and other resources, establishing a priority for the process (in multi-tasking systems), loading program code into memory,

and executing the program. The program then interacts with the user and devices performing its intended function.

Exercise 18. *Say whether the statements below are true or false. Correct the false ones.*

1. An operating system is an interface between the application software and system software. 2. Operating systems offer a number of services to application programs and users. 3. Applications access these services through APIs or system calls. 4. Users may also interact with the operating system with some kind of software UI like typing commands by using CLI or using a graphical user interface. 5. For hand-held and desktop computers, the user interface is generally considered a part of the hardware. 6. On large multi-user systems like UNIX and UNIX-like systems, the user interface is implemented as an application program that runs outside the operating system. 7. The operating system acts as an interface between an application and the hardware. 8. The operating system is a set of instructions which simplifies the development of applications. 9. Writing a program involves the creation of a process by the operating system. 10. The kernel creates a process by assigning memory and other resources, establishing a priority for the process (in multi-user systems), loading program code into memory, and executing the program. 11. The program interacts with the user and devices performing its intended function.

Exercise 19. *Read and translate text 3.*

Text 3. Microsoft Windows

Microsoft Windows is a family of proprietary operating systems that originated as an add-on to the older MS-DOS operating system for the IBM PC. Modern versions are based on the newer Windows NT kernel that was originally intended for OS/2. Windows runs on x86, x86-64 and Itanium processors. Earlier versions also ran on the DEC Alpha, MIPS, Fairchild (later Intergraph) Clipper and PowerPC architectures (some work was done to port it to the SPARC architecture).

As of June 2008, Microsoft Windows holds a large amount of the worldwide desktop market share. Windows is also used on servers, supporting applications such as web servers and database servers. In recent years, Microsoft has spent money on significant marketing, research and

development to demonstrate that Windows is capable of running any enterprise application, which has resulted in consistent rice/ performance records and significant acceptance in the enterprise market.

The most widely used version of the Microsoft Windows family is Windows XP, released on October 25, 2001. In November 2006, after more than five years of development work, Microsoft released Windows Vista, a new version of Microsoft Windows family which contains a large number of new features and architectural changes. Chief among these are a new user interface and visual style called Windows Aero, a number of new security features such as User Account Control, and a few new multimedia applications such as Windows DVD Maker. A server variant based on the same kernel, Windows Server 2008, was released in early 2008.

Windows 7 is the recent version which has recently been developed and released.

Exercise 20. *Say whether the statements below are true or false. Correct the false ones.*

1. Modern versions of Microsoft Windows are based on Windows NT kernel that was originally intended for OS/1. 2. In recent years, Macintosh has spent money on significant marketing, research and development to demonstrate that Windows is not capable of running any enterprise application. 3. The most popular version of the Microsoft Windows family is Windows XP, released on October 15, 2001. 4. In November 2006 Microsoft released Windows Vista, a new version of Microsoft Windows family which contains a large number of new features and architectural changes. 5. Windows 7 is currently under development.

Exercise 21. *Make up questions to the italicized parts of the sentences.*

1. Such applications include ***some small embedded systems***. 2. Operating system can be categorized ***by technology, ownership, licensing, working state, usage, and by many other characteristics***. 3. Modern versions are based ***on the newer Windows NT kernel***. 4. In recent years, ***Microsoft*** has spent significant money on marketing, research and development to demonstrate that Windows is capable of running any enterprise application. 5. The most widely used version of the Microsoft Windows family is ***Windows XP***, released on October 25, 2001.

Exercise 22. *Fill in the blanks with prepositions.*

1. Microsoft Windows is a family … proprietary operating systems that originated as an add-on … the older MS-DOS operating system … the IBM PC. 2. Modern versions are based … the newer Windows NT kernel that was originally intended … OS/2. 3. Windows runs … x86, x86-64 and Itanium processors. 4. Earlier versions also ran … the DEC Alpha, MIPS, Fairchild (later Intergraph) Clipper and PowerPC architectures. 5. … recent years, Microsoft has spent money … significant marketing, research and development to demonstrate that Windows is capable … running any enterprise application, which has resulted … consistent rice/ performance records and significant acceptance … the enterprise market. 6. The most widely used version … the Microsoft Windows family is Windows XP, released … October 25, 2001. 7. … November 2006, … more than five years … development work, Microsoft released Windows Vista, a new version … Microsoft Windows family which contains a large number … new features and architectural changes.

Exercise 23. *Change the following words into adjectives. Use the following suffixes -able; -ible; -al; – ant; -ent; – ful;- ic*; *-ive*; *-ous*; *– y*.

Purpose, reason, exhaust, success, predominance, skill, power, vary, magnet, create, stick, hierarchy, program, comprehend, observe, emerge, gloom, courage, science, compare, noise, form, apply, event, experiment, base, collect, economy, gloom, construct, transmit, function, redundancy, vision, resistance, depend, select, structure, efficiency.

Exercise 24. *Form negative adjectives using the following prefixes un-; in-; ir-; im- ;il-; dis-*; non-.

Reasonable, relevant, satisfied, authorized, complete, fortunate, regular, possible, countable, flammable, respectful, forgettable, direct, perfect, reliable, sensitive, legal, existent, equal, responsible, patient, stable, rational, loyal, honest, restrictive, intentional, decent, proper, skilled, capable, legitimate, metallic, productive, correct, logical.

Exercise 25. *Give the comparative and the superlative degree of the following adjectives.*

Fast, large, intelligent, convenient, complex, simple, small, good, little, much, narrow, long, bad, far, busy, important, regular, reliable, efficient.

Exercise 26. *Decide whether the italicized words are adjectives or adverbs. Translate the sentences.*

1. Computer hardware has undergone ***significant*** improvements over its history. 2. Software security practices should contain ***significantly fewer*** exploitable weaknesses. 3. Malware is a ***general*** term used by computer professionals to mean various annoying software. 4. For handheld and desktop computers the user interface is ***generally*** considered a part of the operating system. 5. A ***typical*** computer includes hardware and software. 6. A database consists of an organized collection of data for one or ***more*** uses, ***typically*** in digital form. 7. The professor delivered ***more*** lectures than his colleague. 8. ***Little*** attention to documentation and ***large*** focus on ***effective*** communication will help produce desired project results. 9. The software should be designed to guard against both ***likely*** and ***unlikely*** events. 10. Students would study ***better*** if they had ***better*** equipment in the classrooms.

Exercise 27. *Choose the right word from those in brackets. Translate the sentences.*

1. The software construction is (close, closely) linked to the software configuration management. 2. Multi-user operating systems allow many (different, differently) users to take advantage of the computer resources (simultaneous, simultaneously). 3. (Fortunate, fortunately), such large-scale methods are not the only kind available for modelling software. 4. Microsoft Windows has a (significant, significantly) majority of market share in the desktop and notebook computer markets, while servers (general, generally) use Linux or other Unix-like systems. 5. A good operating system helps in loading of programs and data from (external, externally) sources into the (internal, internally) memory. 6. A high-level design does not (necessary, necessarily) represent the architecture of the software. 7. Antivirus software may include the ability to (periodic, periodically) receive virus definition updates in order to maintain the software effectiveness. 8. (Internal, internally) architecture is concerned with cost, performance, scalability and other operational matters. 9. A management information system generates information (accurate, accurately) and (regular, regularly). 10. Passive matrix displays contain a grid of (horizontal, horizontally) and (vertical, vertically) wires with an LCD element at each intersection. 11. Hypermedia can be considered

one (particular, particularly) multimedia application. 12. (Common, commonly) methods of data masking include: encryption, decryption, masking and substitution.

Exercise 28. *Translate the word combinations below into Ukrainian.*

To be based on his experience in the MULTICS project; a large, complex family of inter-related operating systems; to resemble the original UNIX; to refer to a large set of operating systems; a diverse group of; a trademark of The Open Group; to conform to standards; a wide variety of machine architectures; for serves as well as for workstations; in academic and engineering environments; market share statistics; to make usage under-represented; to acquire multiple OS; to be applied to; to be originally created for.

Exercise 29. *Read and translate text 4.*

Text 4. UNIX and UNIX-like Operating Systems

Ken Thompson wrote B, which he used to write UNIX, based on his experience in the MULTICS project. B was replaced by C, and UNIX developed into a large, complex family of interrelated operating systems which have been influential in every modern operating system.

The UNIX-like family is a diverse group of operating systems, with several major sub-categories including System V, BSD, and Linux. The name "UNIX" is a trademark of The Open Group which licenses it for use with any operating system that has been shown to conform to their standards. "UNIX-like" is commonly used to refer to a large set of operating systems which resemble the original UNIX.

UNIX-like systems run on a wide variety of machine architectures. They are heavily used for servers in business, as well as for workstations in academic and engineering environments. Free software UNIX versions, such as GNU, Linux and BSD, are popular in these areas.

Market share statistics for freely available operating systems is usually inaccurate since most free operating systems are not purchased, making usage under-represented. On the other hand, market share statistics based on total downloads of free operating systems is often inflated, as there is no economic disincentive to acquire multiple operating systems so users can download multiple systems, test them, and decide which they like best.

Some UNIX versions like HP's HP-UX and IBM's AIX are designed to run only on that vendor's hardware. Others, such as Solaris, can run on multiple types of hardware, including x86 servers and PCs. Apple's Mac OS X, a hybrid kernel-based BSD version derived from NEXTSTEP, Mach, and FreeBSD, has replaced Apple's earlier (non-UNIX) Mac OS.

UNIX interoperability was introduced by establishing the POSIX standard. The POSIX standard can be applied to any operating system, although it was originally created for various UNIX versions.

Notes:

Download – завантажений файл; закладати (*у конструкцію, у програму*); завантажувати, пересилати (*лінією зв'язку*)

discentive – розхолоджування; гальмо, придушення стимулу

interoperability – функціональна сумісність, можливість взаємодії (*програмних і апаратних виробів різних поставників*)

Exercise 30. *Find in text 4 the English for:*

група різних операційних систем; взаємопов'язані операційні системи; відповідати стандартам; академічне та технічне середовище; питома вага на ринку; бути неточним; економічні перешкоди; купувати різноманітні операційні системи; нагадувати вихідний UNIX.

Exercise 31. *Complete the sentences choosing the proper word combinations. Translate them.*

1. Unix developed into a large, complex family of interrelated operating systems which have been influential in … .
   a) every complicated operating system
   b) every modern operating system
   c) every expensive operating system
2. The name "UNIX" is a trademark of … .
   a) the institution for protection of the environment
   b) the Closed Group
   c) the Open Group
3. "UNIX-like" is commonly used to refer to … .
   a) a large set of operating systems
   b) a large set of applications
   c) a large set of interfaces

4. UNIX-like systems run on a wide variety of … .
   a) machine architectures
   b) machinery
   c) machine exhibitions
5. Market share statistics for freely available operating systems is … .
   a) usually inaccurate
   b) inaccurate to some extent
   c) quite inaccurate
6. There is no economic disincentive to acquire multiple operating systems so users can … .
   a) test multiple applications and start operation
   b) download multiple systems, test them, and decide which they like best
   c) check multiple systems, and decide which of them they can sell
7. Some UNIX variants like HP's HP-UX and IBM's AIX are designed to run only … .
   a) on hardware or software
   b) on that vendor's software
   c) on that vendor's hardware
8. Others, such as Solaris, can run on multiple types of hardware, including … .
   a) interface and PCs
   b) x86 servers and PCs
   c) servers and OS
9. UNIX interoperability was introduced by … .
   a) establishing the POSIX standard
   b) acquiring the POSIX versions
   c) granting a license to the POSIX standard
10. The POSIX standard can be applied to … .
   a) any operating system
   b) Mac OS
   c) HP's HP-UX and IBM's AIX

Exercise 32. *Put some key questions on text 4.*

Exercise 33. *Write out all Non-Finite forms of the verbs from text 4. State their forms and functions.*

Exercise 34. *Translate the word combinations below into Ukrainian.*

A line of proprietary, graphical operating systems; all currently shipped Macintosh computers; the successor to; to be built on the technology;

to be first released; a desktop-oriented version; to be usually referred to; the server edition; to be architecturally identical to; its desktop counterpart; to include work group management and administration software tools; to provide simplified access to; a mail transfer agent; a domain name server.

Exercise 35. *Read and translate text 5.*

Text 5. Mac OS X

Mac OS X is a line of proprietary, graphical operating systems developed, marketed, and sold by Apple Inc. loaded on all currently shipped Macintosh computers. Mac OS X is the successor to the original Mac OS, which had been Apple's primary operating system since 1984. Unlike its predecessor, Mac OS X is a UNIX operating system built on technology that had been developed at NeXT through the second half of the 1980s and up until Apple purchased the company in early 1997.

The operating system was first released in 1999 as Mac OS X Server 1.0, with a desktop-oriented version Mac OS X v10.0 following in March 2001. Since then, five more distinct "end-user" and "server" editions of Mac OS X have been released, the most recent being Mac OS X v10.5, which was first made available in October 2007. Releases of Mac OS X are named after big cats; Mac OS X v10.5 is usually referred to by Apple and users as "Leopard".

The server edition, Mac OS X Server, is architecturally identical to its desktop counterpart but usually runs on Apple's line of Macintosh server hardware. Mac OS X Server includes work group management and administration software tools that provide simplified access to key network services, including a mail transfer agent, a Samba server, an LDAP server, a domain name server, and others.

Exercise 36. *Compose five key questions of different kinds (general, special, alternative, disjunctive and subject) on text 5.*

Exercise 37. *Say whether the statements below are true or false. Correct the false ones.*

1. Mac OS has a long history since it has come to the modern word of computer technologies. 2. Mac OS X is a line of proprietary, graphical operating systems developed, marketed, and sold by Cherry Inc., the

latest of which is pre-loaded on all currently shipped Macintosh computers. 3. Mac OS X is a UNIX operating system built on technology that had been developed at NeXT through the second half of the 1980s and up until Apple purchased the company in 1997. 4. The operating system was first released in 1999 as Mac OS X Server 1.0, with a desktop-oriented version (Mac OS X v10.0) following in March 2001. 5. The server edition, Mac OS X Server, is identical to its business counterpart but usually runs on Apple's line of Macintosh server hardware. 6. Mac OS X Server includes work group management and administration hardware tools that provide simplified access to key network services.

Exercise 38. *Compose a dialogue on "Mac OS X", using the word combinations given below. Mind the grammar form.*

A real-time operating system, to include some small embedded systems, to be categorized by a technology, a large amount of the worldwide desktop market share, in the enterprise market, after more than five years of development work, to contain a large number of new features and architectural changes.

Exercise 39. *Find some additional information and speak on:*

1. The difference between system software and application software.
2. Graphical User Interface.
3. The most popular operating systems and the difference between them.
4. Why is Windows so popular?

UNIT 5. SOFTWARE ARCHITECTURE

Exercise 1. *Study the basic vocabulary.*

a) terms

performance – 1) робота, функціонування; експлуатаційні властивості; 2) ККД
global control structure – структура глобального керування
scaling – масштабування, масштабне перетворення
partitioning – розподіл, декомпозиція
typed object – типізований об'єкт
interface point – інтерфейсний вузол

gross-level component – макрорівневий компонент
decompose – розкладати на складові, піддавати декомпозиції
decompositional – декомпозиційний
throughput – продуктивність; пропускна здатність
latency – час очікування, затримка
maintainability – 1) відновлюваність; 2) зручність супроводу *(програмного забезпечення)*
platform independence – незалежність від платформи

b) nouns
issue – проблема, питання
synchronization – синхронізація
composition – 1) склад; 2) побудова, формування, утворення
mapping – відображення
semantics – семантика
extent – ступінь, міра

c) verbs
go beyond – виходити за межі
specify – уточнювати, деталізувати, конкретизувати
emerge – виникати, поставати
aggregate – об'єднувати
eliminate – видаляти; усувати

d) adjectives
gross – великий, макроскопічний
refinable – що підлягає деталізації

e) adverbs
explicitly – 1) ясно, чітко; 2) явно

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Development, refer, structural, environment, architectural, constraint, refinable, object, satisfy, maintainability, protocol, increase, objection, emerge, semantics, overall, integration, discrete, template, share, independence, important, orientation, explicitly, network.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) design, consist, importance, decomposition, computation, specify, overall, selection, interaction, collection, eliminate, aggregate, issue, gross property;

b) compose, overlapping, explicitly, dependent, variable, connect, concrete, important, refinable, emerge, complexity, abstract.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: vary – variant – variety – various – variously

Vary, develop, communicate, assign, maintain, inform, represent, function, part, depend, compose, connect, perform, explicit, define, exist, configure.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

Means of communication between modules; representation of shared information; to go beyond the algorithms and data structures, to emerge as a new kind of problem; assignment of functionality to design elements; scaling and performance; partitioning strategy; discrete, non-overlapping parts; a set of well-formedness constraints that must be satisfied by any architecture; to be decomposed, aggregated, or eliminated in a concrete architecture; key design issues; life-cycle issues such as maintainability, extent of reuse, and platform independence.

Exercise 6. *Read and translate text 1.*

Text 1. Software Architecture Strategies and Concepts

Software architecture is the study of the large-scale structure and performance of software systems. Important aspects of a system's architecture include the division of functions among system modules, the means of communication between modules, and the representation of shared information.

As the size and complexity of software systems increase, the design problem goes beyond the algorithms and data structures of the computation: designing and specifying the overall system structure emerges as a new kind of problem. Structural issues include gross organization and

global control structure; protocols for communication, synchronization, and data access; assignment of functionality to design elements; composition of design elements; scaling and performance; and selection among design alternatives. This is the software architecture level of design.

The architecture consists of (a) a partitioning strategy and (b) a coordination strategy. The partitioning strategy leads to dividing the entire system in discrete, non-overlapping parts or components. The coordination strategy leads to explicitly defined interfaces between those parts.

Software architecture is represented using the following concepts: *1. Component*: An object with independent existence, e.g., a module, process, procedure, or variable. *2. Interface*: A typed object that is a logical point of interaction between a component and its environment. *3. Connector*: A typed object relating interface points, components, or both. *4. Configuration*: A collection of constraints that wire objects into a specific architecture. *5. Mapping*: A relation between the vocabularies and the formulas of an abstract and a concrete architecture. The formula mapping is required because the two architectures can be written in different styles. *6. Architectural style*: A style consists of a vocabulary of design elements, a set of well-formedness* constraints that must be satisfied by any architecture written in the style, and a semantic interpretation of the connectors. Components, interfaces, and connectors are treated as first-class objects- i.e., they have a name and they are refinable. Abstract architectural objects can be decomposed, aggregated, or eliminated in a concrete architecture. The semantics of components is not considered part of architecture, but the semantics of connectors is.

Software architecture is an important level of description for software systems. At this level of abstraction key design issues include gross-level decompositional components, protocols of interaction between those components, global system properties (such as throughput and latency), and life-cycle issues (such as maintainability, extent of reuse, and platform independence).

Exercise 7. *Find in text 1 the English for:*

дослідження великомасштабної структури та функціонування програмних систем; розподіл функцій між системними модулями; обсяг та складність програмних систем; проектування та деталізація загальної структури системи; новий тип завдання; організація на

* well-formedness – формальна правильність

макрорівні та структура глобального керування; протоколи зв'язку, синхронізації та доступу до даних; склад структурних компонентів; вибір між варіантами проектів; стратегії розподілу та координування; розподіл системи на дискретні частини або елементи, що не перетинаються; чітко визначені інтерфейси між частинами; незалежний об'єкт; модуль, процес, процедура, або змінна; логічна сутність взаємодії між елементом та його середовищем; сукупність обмежень, що пов'язують об'єкти в певну архітектуру; абстрактна та конкретна архітектури; відображення формул; формальні обмеження; ключові питання проектування; декомпозиційні компоненти на макрорівні; глобальні властивості системи; пропускна здатність і час очікування.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. Software architecture is the study of the small-scale and large-scale structures and performance of software systems. 2. Important aspects of a system's architecture include the division of functions among system modules, the means of communication between modules, and the representation of shared information. 3. The size and complexity of software systems decrease. 4. The design problem goes beyond the algorithms and data structures of the computation: designing and specifying the overall system structure emerges as a new kind of problem. 5. The architecture consists of (a) a partitioning strategy and (b) a communication strategy. 6. The partitioning strategy leads to dividing the entire system in discrete, overlapping parts or components. 7. The coordination strategy leads to explicitly defined interfaces between those parts. 8. Interfase is a typed object that is a logical point of interaction between components. 9. The formula mapping is required because the two architectures can be written in similar styles. 10. An architectural style consists of a vocabulary of design elements, a set of well-formedness constraints that must be satisfied by any architecture written in the style, and a semantic interpretation of the connectors. 11. Components, interfaces, and connectors are treated as first-class objects – i.e., they have a name and they are nonrefinable. 12. Abstract architectural objects can be decomposed, aggregated, or eliminated in a concrete architecture. 13. The semantics of components is considered part of architecture, as well as the semantics of connectors.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to include	a) a new kind of problem
2) to write	b) a partitioning strategy
3) to go beyond	c) formula mapping
4) to consist of	d) explicitly defined interfaces
5) to emerge as	e) division of functions
6) to lead to	f) part of architecture
7) to require	g) the algorithms and data structures
8) to be treated as	h) constraints
9) to be considered	i) in different styles
10) to satisfy	j) first-class objects

Exercise 10. *Fill in the blanks with prepositions* ***in, to, between, among, beyond, of, as, by, at, for*** *where necessary.*

1. Software architecture is the study … the large-scale structure and performance … software systems. 2. Important aspects … a system's architecture include the division … functions … system modules, the means … communication … modules, and the representation … shared information. 3. As the size and complexity … software systems increase, the design problem goes … the algorithms and data structures … the computation: designing and specifying the overall system structure emerges … a new kind … problem. 4. Structural issues include gross organization and global control structure; protocols … communication, synchronization, and data access; assignment … functionality … design elements; composition … design elements; scaling and performance; and selection … design alternatives. 5. The partitioning strategy leads … dividing the entire system … discrete parts. 6. Interface is a typed object that is a logical point … interaction … a component and its environment. 7. Architectural style consists … a vocabulary … design elements, a set … well-formedness constraints that must be satisfied … any architecture written … the style, and a semantic interpretation … the connectors. 8. Software architecture is an important level … description … software systems. 9. … the level … abstraction key design issues include gross-level decompositional components and protocols … interaction … those components.

Exercise 11. *Fill in the blanks with proper terms* ***(configuration, component, mapping, interface, coordination strategy, connector, partitioning strategy, software architecture)*** *to complete the sentences.*

1. _____ is a typed object relating interface points, components, or both. 2. _____is a strategy that leads to dividing the entire system in discrete, non-overlapping parts or components. 3. _____ is an object with independent existence, e.g., a module, process, procedure, or variable. 4. _____ is a collection of constraints that wire objects into a specific architecture. 5. _____ is a relation between the vocabularies and the formulas of an abstract and a concrete architecture. 6. _____ is the study of the large-scale structure and performance of software systems. 7. _____ is a strategy that leads to explicitly defined interfaces between those parts. 8. _____ is a typed object that is a logical point of interaction between a component and its environment.

Exercise 12. *Answer the questions on text 1.*

1. What is software architecture? 2. What do important aspects of a system's architecture include? 3. Why does the design problem go beyond the algorithms and data structures of the computation? 4. What emerges as a new kind of problem? 5. What do structural issues include? 6. What strategies does the architecture consist of? What do they differ in? 7. What concepts is software architecture represented by? 8. What is a component/interface/ connector? 9. What is configuration/ mapping/ an architectural style? 10. What kind of semantics is considered to be a part of architecture? 11. What do the key design issues of software architecture include?

Exercise 13. *Put all possible questions to the sentences below.*

1. Important aspects of a system's architecture include the division of functions among system modules, the means of communication between modules, and the representation of shared information. 2. As the size and complexity of software systems increase, the design problem goes beyond the algorithms and data structures of the computation. 3. Structural issues include gross organization and global control structure; protocols for communication, synchronization, and data access; assignment of functionality to design elements; composition of design elements; scaling and performance; and selection among design alternatives.

4. The architecture consists of a partitioning strategy and a coordination strategy. 5. The partitioning strategy leads to dividing the entire system in discrete, non-overlapping parts or components. 5. Software architecture is represented using the following concepts: a component, an interface, a connector, a configuration, mapping, an architectural style. 6. Components, interfaces, and connectors are treated as first-class objects. 7. Abstract architectural objects can be decomposed, aggregated, or eliminated in a concrete architecture. 8. The semantics of components is not considered part of architecture, but the semantics of connectors is.

Exercise 14. *Translate into English.*

1. Архітектура програмного забезпечення – це дослідження великомасштабної структури та функціонування програмних систем. 2. Важливі аспекти архітектури системи включають розподіл функцій між системними модулями, засоби зв'язку між модулями та спосіб представлення розподіленої інформації. 3. У зв'язку зі зростанням обсягу та складності програмних систем завдання проектування виходить за межі алгоритмів та структур даних обчислення – проектування та деталізація загальної структури системи постає як новий тип завдання. 4. Архітектурний рівень проектування програмного забезпечення включає такі структурні питання, як організація на макрорівні та структура глобального керування; протоколи зв'язку, синхронізації та доступу до даних; розподіл функціональності між структурними компонентами та їх склад; масштабування та функціонування; вибір варіанту проекту. 5. Архітектура складається зі стратегій розподілу та координування. 6. В архітектурі програмного забезпечення використовують такі поняття: компонент, інтерфейс, з'єднувальний елемент, конфігурація, відображення та архітектурний стиль. 7. Компонент – це незалежний об'єкт, наприклад, модуль, процес, процедура, або змінна. 8. Інтерфейс – це типізований об'єкт, що є пунктом здійснення взаємодії між компонентом та його середовищем. 9. З'єднувальний елемент є типізованим об'єктом, який пов'язує між собою інтерфейсні вузли, компоненти, або й те, й друге. 10. Конфігурація – це сукупність обмежень, що пов'язують об'єкти в певну архітектуру. 11. Відображення відтворює відношення між словниками та формулами абстрактної та конкретної архітектур. 12. Архітектурний стиль складається зі словника структурних компонентів, низки формальних обмежень, яким має

відповідати будь-яка архітектура семантичної інтерпретації з'єднувальних елементів, написана в певному стилі. 13. Компоненти, інтерфейси та з'єднувальні елементи розглядають як об'єкти першого класу, тобто вони мають імена і підлягають деталізації. 14. Абстрактні архітектурні об'єкти можна розкладати, об'єднувати та видаляти в конкретній архітектурі. 15. Архітектура програмного забезпечення є важливим рівнем опису програмних систем. 16. На цьому рівні абстрагування ключові питання проектування включають макрорівневі декомпозиційні компоненти, протоколи взаємодії між цими компонентами, глобальні властивості системи (такі, як: пропускна здатність і час очікування) та питання життєвого циклу (такі, як: зручність супроводу, частотність повторного використання і незалежність від платформи).

Exercise 15. *Write a summary of the text "Software Architecture".*

Exercise 16. *Study the vocabulary to text 2.*

Structural view – структурне представлення
systemic view – системне представлення
component view – компонентне представлення
deployment view – представлення розміщення *(представлення системної архітектури, що виділяє вузли, котрі формують апаратну топологію системи)*
network-centric deployment view – мережецентричне представлення розміщення
logical view – логічне представлення
conceptual view – концептуальне представлення
domain – (*предметна*) галузь, контекст *(середовище, у якому повинна працювати програма)*
legitimate – розумний, прийнятний, обґрунтований
with respect to – відносно
criteria (*pl. від* criterion) – критерії
structural arrangement – структура
evolution – 1) розвиток; 2) розроблення
essential characteristics – істотні, важливі характеристики
reveal – показувати, розкривати
implementation details – деталі реалізації
specifically – а саме, зокрема, конкретно

responsibility – відповідальність
deploy – розміщувати
deployment – розміщення
use case – варіант використання
actor – *прогр.* актор, учасник *(у мові UML – людина або пристрій, що взаємодіє з системою; зображають у вигляді фігурки людини)*
stakeholder – *прогр.* учасник
meet the requirements – відповідати вимогам
development team – група розробників
hardware team – група розробників апаратного забезпечення
installation team – група встановлення
in terms of – в показниках, в одиницях, в перерахунку (*на щось*)
simplistic – спрощений
concise – короткий, стислий, лаконічний
generic – узагальнений
two-tier, three-tier, multi-tier – дворівневий, трирівневий, багаторівневий
layered – багаторівневий, багатошаровий
Component Object Model (COM) – модель компонентних об'єктів Microsoft
Common Object Request Broker Architecture (CORBA) – загальна архітектура брокера / посередника запитів до об'єктів, стандарт CORBA *(технологія побудови розподілених об'єктних програм, запропонована фірмою IBM)*
Microsoft Transaction Server (MTS) – сервер транзакцій корпорації Microsoft
Enterprise Java Beans (EJB) – специфікація EJB *(на серверній частині стандартизує доступ до баз даних та до систем оброблення транзакцій, що важливо для корпоративних прикладних програм, оскільки забезпечує їх перенесення на інші платформи)*
domain specific – що визначається галуззю застосування
account for – пояснювати
map – відображати(ся)
threading – організація потокового оброблення *(повідомлень або даних)*
concurrency – паралелізм
capture – збирати *(дані)*

Exercise 17. *Read and translate text 2.*

Text 2. Essential Characteristics of Software Architecture

Software architecture usually refers to some combination of structural views of a system, with each view being a legitimate abstraction of the system with respect to certain criteria, that facilitate a particular type of planning or usage.

- Software architecture represents the structure of the software. This includes the structural arrangements of software components, and various static and dynamic interrelationships between these components.
- Software architecture is expressed using certain views, each of which serves a specific purpose. Each view is a specific abstraction of the architecture, for a specific purpose.
- Software architecture includes the principles behind design and evolution of the software.

The following are some of the essential characteristics of architecture.

- Software architecture should represent a high-level view of the system revealing the structure, but hiding all implementation details. Specifically, it should reveal attributes such as responsibilities (of the constituents of the architecture), distribution, and deployment.
- Architecture should realize all the use case scenarios. While the use case model serves to record the functional requirements as seen by various actors, the architecture should enable the stakeholders of the software to walk through the scenarios of each use case. This guarantees that the structure as represented by the architecture meets the functional requirements.
- It should present other systemic views to all the stakeholders of the software. Examples are – a component view for the development team, a network-centric deployment view for the network and hardware team, and a distribution-centric deployment view for the installation team etc.

However, how does the architecture look like? Some of the common representations of software architecture are as follows. As discussed below, none of these representations is complete.

- **High-level design:** A simplistic approach is to represent the architecture as a concise view of a high-level design of the software. However, design is an implementer's view of the software – a view that reveals how to arrive at the structure of the software. So a high-level design does not necessarily represent the architecture of the software.

- **Deployment:** This is the most common form of representations of architecture. In this view, the software is described in terms of how it is deployed across various platforms, and how these parts communicate with each other. Note that deployment view is only one of the possible views of architecture, and does not necessarily reveal the structure of the software. Also note that during the life-cycle of a software, the deployment could change with no or minimal changes to the structure of the software. This is one of the reasons that is driving distributed component based technologies.
- **Generic Technology Architectures:** In this form, software architecture is represented as a two-tier, three-tier or a multi-tier system. Note that architectures such as these, distributed (and layered) architectures such as COM or CORBA, or component based architectures such as MTS or EJB are generic and do not address the needs of the domain in which the software is to operate and evolve. However, these technology architectures provide the basis for developing domain specific application architectures.

The architecture should at least present the following views of the software (in the order of the importance).

- **Logical or Conceptual View:** This view of the software represents various abstractions of the system and accounts for various use case scenarios. Using this view, one should be able to walk through these abstractions to realize the use case scenarios. In the case of distributed applications, some of these abstractions may directly map to distributed components.
- **Deployment View:** As pointed earlier, this view depicts how various parts of the software are deployed.

In both views, it is necessary to depict responsibilities of each of the parts of the architecture, static and/or dynamic dependencies between them, the nature of communication between the parts, etc. Additional views such as development views (to show how the application will be developed) and process views (to reveal threading, concurrency etc.) may also be considered if required.

In general, software architecture is considered important as it serves as a means of mutual communication between the stakeholders of the software, allows to capture early design decisions, and lets the architecture be reused for similar systems in the same domain.

Exercise 18. *Answer the questions on text 2.*

1. What does software architecture usually refer to? 2. What does the structure of the software include? 3. How is software architecture expressed? 4. What are some of the essential characteristics of software architecture? 5. What are examples of systemic views presented to all stakeholders of the software? 6. What does a high-level design represent? 7. What does a deployment view show? 8. How is software architecture represented in generic technology architectures? 9. What are examples of generic technology architectures? 10. What does a logical or conceptual view present? 11. What is it necessary to depict in logical and deployment views? 12. What additional views may also be considered? 13. Why is software architecture considered important?

Exercise 19. *Choose the right form of the verbs in brackets. Mind the sequence of tenses.*

1. He says, "I … just … a class of problems!" (has solved; have solved; had solved). 2. They told us, "We … already …a high-level prototype". (have built; had built; has built). 3. The students say, "We …just …proven patterns to solve problems". (had used; were used; have used). 4. He said, "I … … … the system in compliance with the plan by next month". (will have developed; would have developed). 5. They announce, "We … … our decisions to developers". (have disseminated; had disseminated; will be disseminated). 6. He told us, "I … … … structural patterns by evening". (would have identified; will have identified). 7. She said, "Application developers … already … available capabilities". (has used; had used; have used). 8. We … … that he … just … the problems to be solved. (find out/have identified; found out/had identified; found out/has identified). 9. He asked me what … … me facilitate the standardization of services. (has helped; had helped; will help). 10. I didn't know you … … your assignment. (have completed; had completed; has completed).

Exercise 20. *Use the proper tense form and voice of the verbs in brackets.*

1. The complexity of software systems (increase) significantly recently. 2. A new kind of problem (appear) before the designing the overall system structure. 3. By next year two architectures (write) in different styles. 4. Abstract architectural objects just (decompose).

5. He already (represent) the architecture as a concise view of a high-level design of the software. 6. The deployment view just (reveal) the structure of the software. 7. I (finish) presenting the view of the software by next week. 8. The evolution of software (review) before his arrival. 9. By next week he (capture) design decisions. 10. You already (determine) the components to build the system? 11. He just (realize) the importance of past experiences for software architecture. 12. How long you (solve) this technical problem? 13. He (gain) enough breadth and depth in the relevant domain long before I did it. 14. My friend (not have) the capability to envision a software system before it was developed. 15. I hope you already (understand) the importance of software architects. 16. You (get) enough development and debugging skills by next interview? 17. Unfortunately many projects already (make) mistake of trying to impose a single partition in multiple component domains. 18. His irresponsible actions (lead) to problems in development before we had time to correct the mistakes. 19. The implementation of a complex functional feature (split) between two groups by next meeting. 20. Yesterday I knew that the performance (compromise). 21. Since recently we (use) the resulting models to plan the subsequent development activities. 22. You (determine) the purpose and specifications of software before you started developing a plan for a solution? 23. Software developers (design) a plan by next decade? 24. He (not analyze) yet the software requirements. 25. It just (let) us produce various models. 26. Software designers (evaluate) these models since last month. 27. He said that various alternative solutions and trade-offs (examine) before. 28. More sophisticated methods (apply) by next year. 29. We knew that a set of fundamental design concepts (evolve). 30. I'm sure that a good software architecture already (yield) a good return on investment.

Exercise 21. *Put the verbs in brackets into an appropriate form of Perfect or Indefinite Tense.*

1. Last time he (divide) the program structure both horizontally and vertically. 2. He (work) on designing modules last week. 3. He never (consider) many aspects in the design. 4. When you (manage) to maintain the software effectiveness? 5. Since last trial the components (test). 6. He already (design) the software with a resilience to low memory conditions. 7. You ever (achieve) these goals? 8. When you

(choose) the default values for the parameters? 9. You already (enumerate) all design criteria? 10. Last week he (work) at multiple levels of abstraction. 11.Today I (know) a lot about Human Machine Interface. 12. A month ago he (buy) a new backlit display. 13. He (forget) to ask about data transfer rate when he was in the office. 14. We (study) local area networks since last month. 15. This week they (install) a full keypad. 16. When you (learn) about operating temperature? 17. I don't think he ever (hear) about Ethernet. 18. You already (design) the program? 19. When you last (change) your display? 20. I (learn) about this feature yesterday.

Exercise 22. *Change the sentences into indirect speech.*

1.The teacher asked, "Did you define the problem?" 2. He enquired," Will you have finished your report by evening? 3. The professor told us, "We have already performed the major part of our work". 4. My friend said to me, "I have just separated the interface from implementation". 5. They said to us, "We have been trying to make it work for a long time". 6. I asked my groupmate, "Have you already found your mistake?" 7. He said, "I have implemented the design before you did it." 8. She asked, "How long have you been searching for the decision?" 9. The student said to the teacher," I will have sent my report by Friday." 10. He asked, "Did they understand their tasks last time?"

Exercise 23. *Study the vocabulary to text 3.*

Software architect – розробник структури ПЗ, фахівець з архітектури систем ПЗ
blueprint – *тут:* схема, будова, структура
overview – загальне уявлення, загальна картина
structural architect – архітектор-будівельник
all along – у/впродовж усього часу
edge – перевага
envision – уявляти, передбачати
exposure (to) – *тут:* можливе залучення (*до чогось*)
impart – наділяти, надавати
anticipate – передбачати, передчувати
navigator – штурман, навігатор
shoot *(p., pp.* shot*)* – стріляти
convince – переконувати

cross – мішаний, гібридний
proficiency – досвідченість, уміння, вправність, професійність
appreciation – розуміння, уміння добре розібратися (*у чомусь*)
maturity – розвиненість
predefined – наперед визначений, стандартний
pattern – шаблон
design pattern – конструктивний шаблон
architecture pattern – архітектурний шаблон
vogue – популярність, широке застосування
industry – індустрія, галузь
recurring – такий, що повторюється, періодичний,
excited – захоплений
be on the lookout (*for*) – бути насторожі (*щодо чогось*), пильнувати, шукати (*щось*)
challenging – складний
thought leadership – інтелектуальне лідерство
equate – прирівнювати, ототожнювати
progression – просування, рух уперед, прогрес

Exercise 24. *Translate the word combinations below into Ukrainian.*

To give an edge; envisioning a solution; to impart in you the ability to choose among various solutions available; to anticipate and solve technical problems; to gain enough breadth and depth in the relevant domain and technologies; to convince the people above and below them in the hierarchies; to be a vogue in the Software Development industry; to equate one's success to that of the customer's; to be a natural progression.

Exercise 25. *Read, translate and entitle text 3.*

Text 3

Software architecture is the blueprint of a software system. It provides an overview of the composition and functionality of the given software system. Just like a structural architect, a software architect needs to analyze the requirements, determine components that should be used to build the system, and support the project by guiding and solving problems all along the execution cycle. Architecting software is like planning for war. Past experiences are very useful here. Strategizing gives you an edge. Understanding of the domain provides you with capability to analyze the requirements and envisioning a solution. Expo-

sure to various tools and technologies imparts in you the ability to choose among the various solutions available, and anticipate and solve technical problems.

A person becomes a software architect when he has gained enough breadth and depth in the relevant domain and technologies, and has the capability to envision a software system before it is developed.

Architects are needed for most of the Software Projects, more as the navigator for a sailing ship. Architects design the software system, guide the development team in implementing the system, and anticipate/diagnose problems, find/develop solutions to those problems. These days, most of the organizations are realizing the importance of software architects, because with an architect you are no longer shooting in the dark.

Architects need to have good written/spoken communication since they have to convince the people above and below them in the hierarchies about their ideas effectively, strong development and debugging skills, domain and technology proficiency and appreciation, and customer's point of view can be very useful for an architect.

Due to the maturity of Software Development practice, there are several predefined solutions available for certain problems called Patterns. Design and Architecture patterns are a vogue in the Software Development industry. They are the solutions for a recurring class of problems. Most of the time, they are the best possible solutions to those problems. Knowledge of patterns in their domain/technology areas is an added advantage for an architect.

For a person, who is excited by new tools, technologies and domains, who is on the lookout for challenging problems to solve, who is capable of thought leadership, who equates his success to that of the customer's, the role of Software Architect is a natural progression.

Exercise 26. *Find in text 3 the English for:*

структура програмної системи; загальне уявлення про будову і функціональність програмної системи; інструктування і вирішення проблем впродовж усього циклу виконання; розуміння предметної області; можливе залучення різноманітних засобів і технологій; міцні навички у розробці і налагоджуванні; вправність і уміння добре розібратися як у предметній області, так і в техніці; завдяки розвиненій практиці розроблення програмного забезпечення; стандартні рішення; клас задач, що періодично постають перед розроб-

ником; нові інструменти, технології та сфери застосування; шукати складних проблем, що потребують розв'язання; здатний на інтелектуальне лідерство.

Exercise 27. *Answer the questions on text 3.*

1. What is software architecture? 2. What does software architecture provide? 3. What does a software architect need to do? 4. What aspects should he consider in architecting software? 5. When does a person become a software architect? 6. What is architects' job? 7. What skills do architects need to have? 8. What are patterns? 9. For what kind of person is the role of Software Architect a natural progression? 10. Do you have any experience in architecting software? Speak to your groupmates about it.

Exercise 28. *Explain the meaning of the following words in English.*

Software architecture, a blueprint, a software system, an overview, a software architect, a problem, experience, to give an edge, a domain, to envision, an exposure, a software project, shooting in the dark, debugging, a skill, technology proficiency, appreciation, maturity, predefined solution, a pattern, a vogue, Software Development industry, a recurring class of problems, to be on the lookout for, a challenging problem, a leadership, a success, a customer, a progression.

Exercise 29. *Study the vocabulary to text 4.*

affect – впливати
write time – час запису
build time – час компонування / побудови поточного варіанту програми
configuration time – час конфігурування
upgrade time – час модернізування, оновлення
startup – (за)пуск
start time – час запускання, початковий момент
run time – час виконання (*програми*), час прогону (*програми*)
shutdown – вимкення, зупинка
shutdown time – час зупинки
porting – портування *(у програмуванні портування розуміють як адаптацію програми або її частини до роботи в іншому середовищі, відмінному від того, для якого вона була написана)*

diversification – введення різноманітності, диверсифікація
address – займатися *(проблемою, питанням)*; братися (*за щось*)
primarily – головним чином
appropriate – відповідний, належний
similarly – так само, у той самий спосіб, аналогічно
independent operation – автономне (незалежне) функціонування
failure mode – стан відмови
executable – що виконується
availability – 1) експлуатаційна готовність; 2) працездатність, неперервність
execution thread – потік виконання; потік завдань, що виконуються
equate – *тут:* пов'язувати
in turn – почергово, послідовно, своєю чергою
succeed – досягати мети, мати успіх
adjustment – регулювання, налагодження, коригування
eventually – врешті решт, зрештою, з часом
invariably – неодмінно
occasionally – іноді, час від часу
cluster – сукупність, група, кластер
interprocess communication mechanism – механізм взаємодії між процесами
consideration – міркування
fault tolerance – відмовостійкість
cure – ліки, засіб
subsequently – згодом, пізніше, потім
timing characteristic – часова характеристика
violate – порушувати
cure – засіб вирішення проблеми
timeliness – своєчасність *(реагування, подання інформації в комп. системі)*
capacity – продуктивність, пропускна здатність
conformance (*to*) – відповідність (*до чогось*)
propagation – поширення, передавання
pertinent to sth – такий, що стосується чогось, відповідний
result in – мати результатом (*щось*), закінчуватися (*чимось*)
associated – пов'язаний (*із чимось*), відповідний (*до чогось*)

Exercise 30. *Choose verbs among the following words. Put the first letters of the verbs into the cells in the same order. Read and translate the word.*

Software, include, different, internal, relation, component, architecture, necessitate, domain, transform, exist, porting, availability, with, load, important, independent, specific, architectural, legislate, current, along, scheme, extend, basic, create, therefore, earlier, concerns, partition, transact.

Exercise 31. *Give synonyms for the following words.*

Require, basic, component, partition, complex, execution, in the end, plan, particular, combine, user, modern, along with, significant, address, the problem, cluster, initial, subsequently, specify, pertinent to.

Exercise 32. *Read and translate text 4.*

Text 4. Architectural Structures

The structure of software in a component domain is created by a partition of software into components and their composition into an integrated whole. For every system it is necessary to determine which structures of software affect architecturally significant requirements and to group the requirements in such a way that each group is supported primarily by independent structures that exist in different component domains.

One effective way to identify independent (or partly independent) requirements is by different stages of software life cycle with which they are concerned. A typical (though somewhat simplified) set of stages when different structures of software play major roles includes write time, build time, configuration time, upgrade time, start time, run time, and shutdown time. The most important software structure at write time is the structure of modules. Thus write time-related requirements, such as feature addition and evolution, porting, and diversification, are addressed primarily by appropriate module structures that play a major role at write time.

Similarly, start time-related requirements (such as order, presence, independent operation, and failure modes) are addressed primarily by appropriate executable structures – the startup or shutdown unit or component. In addition, of course, run time-related requirements, such as performance or availability, are addressed by the structures of objects and execution threads – the domain of run-time software components.

Many projects make the mistake of trying to impose a single partition in multiple component domains, such as equating threads with objects, which are equated with modules, which in turn are equated with files. Such an approach never succeeds fully, and adjustments eventually must be made, but the damage of the initial intent is often hard to repair. This invariably leads to problems in development and occasionally in final products.

In one case, implementation of a complex functional feature was split between two groups. Two functional clusters were defined, along with the necessary interfaces. Unnecessarily, the modules also ended up in different processes and had to interact at run time using slower interprocess communication mechanisms.

Another example involved a system that was partitioned into a set of distributed processes. The partition was motivated by considerations of required parallelism, availability, and fault tolerance. This partition was subsequently used to allocate additional functionality, which affected resource requirements and timing characteristics, violating the original design. As a cure, non-real-time functionality was allocated to new components. However, because the software architecture was identified with its process structure, these components became independent processes. Consequently, the components had complex interfaces and performance was compromised.

Designing a software architecture must start with specific architectural concerns, specify the partition in different component domains, along with a scheme for integration and coordination of the parts, and explain how this specific partition and the corresponding integration of the software address the specified architectural concerns. Examples of architectural concerns may include timeliness, capacity, availability, effective division of work, conformance to standards, use of existing parts, or controlled propagation of change. To address these concerns, different partitions may exist in different component domains.

From the point of view of software reuse, architecture that separates concerns pertinent to different requirement and component domains also results in more reusable components. Therefore it is important to recognize multiple software existence planes with the associated component domains and independent partitions of software and their relations to different requirement domains.

Exercise 33. *Put some key questions on text 4.*

Exercise 34. *Say whether the statements below are true or false. Correct the false ones.*

1. A scheme of software in a component domain is created by a partition of software into components and their composition into an integrated whole. 2. One effective way to identify independent requirements is by different stages of software life cycle with which they are concerned. 3. A typical set of stages when different structures of software play major roles includes run time, write time, configuration time, build time, start time, upgrade time and shutdown time. 4. The most important software structure at write time is the structure of diagrams. 5. Many projects make the mistake of trying to impose a larger partition in multiple component domains, such as equating threads with objects. 6. Implementation of a complex functional feature was split between three groups. 7. Two functional clusters were defined, along with the necessary interfaces. 8. The modularities also ended up in different processes and had to interact at run time using slower interprocess communication mechanisms. 9. Because the software architecture was identified with its process structure, these components became independent processes. 10. Examples of component domains may include timeliness, capacity, availability, effective division of work, conformance to standards, use of existing parts, or controlled propagation of change. 11. From the point of view of software architecture, reuse that separates concerns pertinent to different requirement and component domains also results in more reusable components. 12. It is important to recognize multiple software existence planes with the associated component domains and independent partitions of software and their relations to integration of the software.

Exercise 35. *Form all possible word combinations with the words from both columns. Translate them.*

1) to be created by	a) appropriate module structures
2) to affect	b) considerations of required parallelism
3) to play	c) a partition of software into components
4) to be addressed by	d) functional clusters
5) to make	e) major roles
6) to lead to	f) complex interfaces
7. to define	g) more reusable components

8. to be motivated by	h) the mistake
9. to have	i) problems in development
10. to result in	j) architecturally significant requirements

Exercise 36. *Decipher the abbreviations below.*

MTS, EJB,COM, CORBA, VDU, SPARC, API, CLI, UI, USB.

Exercise 37. *Compose a dialogue on "Architectural structures".*

Exercise 38. *Find some additional information and speak on:*

1. Architectural strategies and concepts.
2. Essential characteristics of software architecture.
3. Structure of software in a component domain.

UNIT 6. SOFTWARE DESIGN

Exercise 1. *Study the basic vocabulary.*

a) terms

software design – 1) проектування ПЗ; 2) проект ПЗ
software solution – 1) програмний продукт; 2) програмне рішення
specification – 1) специфікація; 2) часто *pl* технічні вимоги
platform-dependent (platform-specific) – залежний від платформи
software architectural design – архітектурне проектування ПЗ, проектування архітектури ПЗ
software detailed design – детальне проектування ПЗ

b) nouns

activity – *зуг:* операція
level of detail – рівень деталізації
trade-offs – *зуг:* плюси та мінуси, баланс переваг і рівень недоліків
listing – перелік

c) verbs

employ – наймати *(на роботу)*
call for – вимагати
fulfill – виконувати, задовольняти *(вимоги)*
examine – досліджувати, аналізувати
fit between – розташовуватися між, бути на перетині
allow for – дозволяти, забезпечувати

e) adverbs
precisely – точно
sufficiently – достатньо, достатньою мірою

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Precisely, define, concept, evaluate, object, subsequent, maintainability, sufficiently, procedure, fulfill, availability, implement, text, enable, item, examine, blueprint, analyze, integration, various, level, observable, interoperability, sophisticated, tolerance, internal, year.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) software, purpose, architecture, designer, enable, consist, identify, various, precisely, activity, blueprint, trade-off, examine, fulfil, in the end, requirement, basis;

b) hardware, advantage, platform-independent, sufficiently, sophisticated, precisely, finally, top-level design, internal, employ, various.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: specify – specification – specifier – specific – specifically

Specify, solve, process, depend, design, require, describe, produce, develop, precise, add, sufficient, vary, implement, maintain, operate, evaluate, available, refine, elaborate.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

Software design; software solution; to determine the purpose and specifications of software; to employ designers; platform-independent or platform-specific; availability of the technology called for by the design; software requirements; to produce a description of the software internal structure; more precisely; to enable construction; software developing; to plan subsequent development activities; in addition to; a standard listing of soft-

ware life cycle processes; to consist of two activities; to fit between software requirements analysis and software construction; identifying various components; to describe each component sufficiently.

Exercise 6. *Read and translate text 1.*

Text 1. Software Design Activities

Software design is the process of defining the architecture, components, interfaces, and other characteristics of a system or component and planning for a software solution. After the purpose and specifications of software are determined, software developers will design or employ designers to develop a plan for a solution.

Software design may be platform-independent or platform-specific, depending on the availability of the technology called for by the design.

Viewed as a process, software design is the software engineering life cycle activity in which software requirements are analyzed in order to produce a description of the software internal structure that will serve as the basis for its construction. More precisely, software design (the result) must describe the software architecture and the interfaces between those components. It must also describe the components at a level of detail that enables their construction.

Software design plays an important role in developing software: it allows software engineers to produce various models that form a kind of blueprint of the solution to be implemented. We can analyze and evaluate these models to determine whether or not they will allow us to fulfill the various requirements. We can also examine and evaluate various alternative solutions and trade-offs. Finally, we can use the resulting models to plan the subsequent development activities, in addition to using them as input and the starting point of construction and testing.

In a standard listing of software life cycle processes software design consists of two activities that fit between software requirements analysis and software construction:

- Software architectural design (sometimes called top-level design): describing software top-level structure, organization and identifying various components.
- Software detailed design: describing each component sufficiently to allow for its construction.

Exercise 7. *Find in text 1 the English for:*

процес визначення архітектури, компонентів, інтерфейсів та інших характеристик системи; планування програмного продукту; призначення та технічні вимоги до програмного забезпечення; розробник програмного забезпечення; бути незалежним чи залежним від платформи; технологія, потрібна для проектування; операція з розроблення життєвого циклу програмного забезпечення; описувати компоненти на рівні деталізування, що спрощує їх побудову; відігравати важливу роль в розробленні програмного забезпечення; структура рішення, яке потрібно реалізувати; оцінювати моделі; задовольняти різні вимоги; аналізувати та оцінювати різні варіанти рішень; стандартний перелік процесів життєвого циклу програмного забезпечення; проектування архітектури програмного забезпечення; детальне проектування програмного забезпечення.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. Software design is the process of defining the architecture, components, interfaces, and other characteristics of a system or component and planning for a hardware solution. 2. After the purpose and specifications of software are determined, software architects will design or employ designers to develop a plan for a solution. 3. Software design may be platform-independent or platform-specific, depending on the availability of the technology called for by the design. 4. Viewed as a process, software design is the software engineering life cycle activity in which software requirements are analyzed in order to produce a description of the software's internal structure that will serve as the basis for its construction. 5. More precisely, software design (the result) must describe the software construction and the interfaces between those components. 6. Software design plays a minor role in developing software. 7. Software design allows software engineers to produce various models that form a kind of blueprint of the solution to be implemented. 8. In a standard listing of software life cycle processes software design consists of three activities that fit between software requirements analysis and software construction.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to define	a) a plan for a solution
2) to employ	b) software requirements
3) to determine	c) components and interfaces
4) to develop	d) various models
5) to be	e) the purpose and specifications of software
6) to analyze	f) subsequent development activities
7) to produce	g) alternative solutions
8) to serve as	h) designers
9) to examine	i) platform-independent
10) to plan	j) the basis for construction

Exercise 10. *Fill in the blanks with prepositions* ***in, on, of, for, to, as, between, by*** *where necessary.*

1. Software design is the process … defining the architecture, components, interfaces, and other characteristics … a system or component and planning … a software solution. 2. After the purpose and specifications … software are determined, software developers will design or employ designers to develop a plan … a solution. 3. Software design may be platform-independent or platform-specific, depending … the availability … the technology called … … the design. 4. Viewed … a process, software design is the software engineering life cycle activity … which software requirements are analyzed … order to produce a description … the software internal structure that will serve … the basis … its construction. 5. Software design must also describe the components … a level … detail that enables their construction. 6. Software design plays an important role … developing software: it allows software engineers to produce various models that form a kind … blueprint … the solution to be implemented. 7. Finally, we can use the resulting models to plan the subsequent development activities, … addition … using them … input and the starting point … construction and testing. 8. … a standard listing … software life cycle processes software design consists … two activities that fit … software requirements analysis and software construction.

Exercise 11. *Fill in the blanks with proper terms (**component, software architectural design, software development, software design, software detailed design, software construction, interface, software architecture**) to complete the sentences.*

1. _____ is a typed object that is a logical point of interaction between a component and its environment. 2. _____ is the study of the large-scale structure and performance of software systems. Important aspects of a system's architecture include the division of functions among system modules, the means of communication between modules, and the representation of shared information. 3. _____ is the process of defining the architecture, components, interfaces, and other characteristics of a system or component and planning for a software solution. 4. _____is the term that refers to the detailed creation of working, meaningful software through a combination of coding, verification, unit testing, integration testing, and debugging. 5. _____ is describing software's top-level structure, organization and identifying various components. 6. _____ is an object with independent existence, e.g., a module, process, procedure, or variable. 7. _____ is the process of writing and maintaining the source code that may include research, prototyping, modification, reuse, reengineering, maintenance, or any other activities that result in software products. 8. _____ is describing each component sufficiently to allow for its construction.

Exercise 12. *Answer the questions on text 1.*

1. What is software design? 2. What may software design depend on? 3. What kind of activity is software design? 4. What must software design describe? 5. What is the role of software design in developing software? 6. What can software models be used for? 7. What activities does software design consist of in a standard listing of software life cycle processes? 8. What is the difference between these activities? 9. Can you explain the difference between software architectural design and software detailed design?

Exercise 13. *Put all possible questions to the sentences below.*

1. After the purpose and specifications of software are determined, software developers will design or employ designers to develop a plan for a solution. 2. Software design may be platform-independent or plat-

form-specific, depending on the availability of the technology called for by the design. 3. Viewed as a process, software design is the software engineering life cycle activity in which software requirements are analyzed in order to produce a description of the software's internal structure that will serve as the basis for its construction. 4. Software design must describe the software architecture and the interfaces between those components. 5. Software design plays an important role in developing software. 6. Software design allows software engineers to produce various models that form a kind of blueprint of the solution to be implemented. 7. We can analyze and evaluate these models to determine whether or not they will allow us to fulfill the various requirements. 8. In a standard listing of software life cycle processes software design consists of two activities that fit between software requirements analysis and software construction.

Exercise 14. *Translate into English.*

1. Проектування програмного забезпечення – це процес визначення архітектури, компонентів, інтерфейсів та інших атрибутів системи та планування програмного продукту. 2. Після визначення мети та технічних характеристик програмного забезпечення, розробники програмного забезпечення розробляють план створення продукту. 3. Проектування програмного забезпечення може бути незалежним або залежним від платформи, що зумовлено наявністю технології, необхідної для проекту. 4. Якщо проектування програмного забезпечення розглядати як процес, воно є операцією розроблення життєвого циклу програмного забезпечення, де аналізуються вимоги до програмного забезпечення для того, щоб описати його внутрішню структуру, яка слугуватиме основою для конструювання. 5. Точніше кажучи, проект програмного забезпечення має описувати архітектуру програмного забезпечення та інтерфейси між його компонентами. 6. Проектування програмного забезпечення дозволяє інженерам з розроблення програмного забезпечення створювати різні моделі, що являють собою певну схему рішення, яке необхідно реалізувати. 7. Можна проаналізувати та оцінити ці моделі та визначити, чи вони будуть задовольняти різні вимоги. 8. Можна також дослідити та оцінити різні варіанти рішень, їх переваги та недоліки. 9. Зрештою можна використати кінцеві моделі для планування наступних етапів розроблення, а також щоб розпо-

чати конструювання та тестування. 10. У стандартному переліку процесів життєвого циклу програмного забезпечення розроблення програмного забезпечення розпочинається з аналізу вимог до нього та закінчується конструюванням. 11. Проектування архітектури програмного забезпечення включає опис високорівневої структури програмного забезпечення, організацію та визначення різних компонентів. 12. Детальне проектування програмного забезпечення включає опис кожного компонента, достатній для його побудови.

Exercise 15. *Write a summary of the text "Software Design Activities".*

Exercise 16. *Study the vocabulary to text 2.*

Foundation – фундамент, підвалина, основа
sophisticated – складний, витончений
observable – спостережуваний
retain – утримувати, зберігати
relevant – такий, що стосується *(даного питання, справи),* релевантний
refinement – вдосконалення, підвищення якості
elaboration – детальне розроблення, уточнення
stepwise – поетапний
in a fashion – певним чином, певною мірою
complementary – доповняльний, додатковий
modularity – модульність
yield – давати результат
schedule – графік
imply – означати, передбачати, мати на увазі
top down – згори донизу
data structure – структура даних
software procedure – програмна процедура
information hiding – приховування інформації

Exercise 17. *Translate the word combinations below into Ukrainian.*

Design concept; to apply sophisticated methods; a set of fundamental design concepts; an observable phenomenon; in order to retain the information which is relevant to a particular purpose; the process of elaboration; decomposing a macroscopic statement of function in a stepwise fashion; complementary concepts; to yield a good return on

investment with respect to the desired outcome of the project; in terms of performance; quality, schedule and cost; to imply a hierarchy of control; horizontal and vertical partitions; information hiding, to specify modules.

Exercise 18. *Read and translate text 2.*

Text 2. Design Concepts

Design concepts provide a software designer with a foundation from which more sophisticated methods can be applied. A set of fundamental design concepts has evolved. They are:

1. Abstraction. Abstraction is the process or result of generalization by reducing the information content of a concept or an observable phenomenon, typically in order to retain the information which is relevant to a particular purpose.

2. Refinement. It is the process of elaboration. A hierarchy is developed by decomposing a macroscopic statement of function in a stepwise fashion until programming language statements are reached. In each step, one or several instructions of a given program are decomposed into more detailed instructions. Abstraction and refinement are complementary concepts.

3. Modularity. Software architecture is divided into components called modules.

4. Software Architecture. It refers to the overall structure of the software and the ways in which that structure provides conceptual integrity for a system. A good software architecture will yield a good return on investment with respect to the desired outcome of the project, e.g. in terms of performance, quality, schedule and cost.

5. Control Hierarchy. A program structure that represents the organization of program components and implies a hierarchy of control.

6. Structural Partitioning. The program structure can be divided both horizontally and vertically. Horizontal partitions define separate branches of modular hierarchy for each major program function. Vertical partitioning suggests that control and work should be distributed top down in the program structure.

7. Data Structure. It is a representation of the logical relationship among individual elements of data.

8. Software Procedure. It focuses on the processing of each module individually.

9. Information Hiding. Modules should be specified and designed so that information contained within a module is inaccessible to other modules that have no need for such information.

Exercise 19. *Answer the questions on text 2.*

1. What do design concepts provide a software designer with? 2. What are the fundamental design concepts? 3. What is abstraction? 4. What is refinement? 5. How is hierarchy developed? 6. What is software architecture divided into? 7. What is software architecture? 8. What will a good software architecture yield? 9. What is control hierarchy? 10. What does the horizontal and vertical partitioning of a structure suggest? 11. What is data structure? 12. What does a software procedure focus on? 13. What does information hiding imply?

Exercise 20. *Change the following sentences to the Passive Voice.*

1. Recently this firm has designed a new operating system. 2. He said that Sun Microsystems had developed Solaris as a more open option of SunOS. 3. Solaris will have got the largest share of the Internet market by next decade. 4. This new operating system has already offered a number of services to application programs and users. 5. We knew that they had released Windows XP in October 2001. 6. Each field of science will have imposed it's own requirements on the hardware by next year. 7. This kind of hardware has greatly facilitated connections between computers. 8. The teacher told us that the system case had provided a solid structural framework. 9. The student says that he will have presented his project by next month.10. An operating system has relieved applications from having to control the hardware. 11. The teacher said that they designed the Macintosh operating system to be used on Apple Macintosh computers. 12. He will have assigned the tasks by next week. 13. Microsoft has spent money to significant marketing research and development. 14. He said that after more than five years of development work, Microsoft released Windows Vista. 15. By next decade they will have replaced an old version by a new one.

Exercise 21. *Change the following sentences to the Active Voice.*

1. Since the last 25 years numerous distinct activities have been identified by researchers. 2. We found out that detailed design, unit testing, integration testing had been included in construction. 3. Substantial

creativity and judgement have been involved in the process of construction. 4. I suppose that the latest enterprise information systems will have been adapted by some client service companies by next decade. 5. Reduced complexity has been recently achieved through emphasizing the creation of the code that is simple and readable. 6. They said that our tasks had already been defined by him. 7. A lot of efforts will have been made to reach an agreement by next time. 8. Significant constraints have been recently introduced in our activity. 9. I knew that some unanticipated actions had been observed by him. 10. Details of the software design will have been fleshed out by next meeting. 11. Many mathematical models have been classified as: linear and nonlinear. 12. The professor explained that the notations of the object-modelling technique and object-oriented software engineering had been synthesized by UML. 13. A standard modelling language will have been created by us by that time. 14. UML has been designed to be compatible with the leading object-oriented software development methods. 15. We learnt that they had been used extensively to describe the functionality of software system.

Exercise 22. *Make the following sentences interrogative using the Passive Voice.*

Model: Employees have already used new programs. – Have new programs been already used by employees?

1. He had illustrated his example before we asked him about it. 2. We will have designed a new program before they know it. 3. We have identified primary keys very quickly. 4. We have implemented data model by generating SQL. 5. They said that they had used the process model in structured analysis and design methods. 6. He will have carried out these experiments in his laboratory by next month. 7. We have used the model due to its ability to express concurrency. 8. He has used the task model to create high-level system. 9. They will have attached all components by the time you ask. 10. This device has performed all calculations at high speed. 11. They will have introduced significant changes by next week. 12. I knew that he had already determined the types of peripherals. 13. We will have run applications on the machine before they come. 14. He has downloaded the programs quite easily. 15. She has bought a new lightweight notebook to carry on business trip.

Exercise 23. *Complete these sentences using the correct passive form of the verbs in brackets (Present, Past or Future Perfect Passive).*

1. The teacher said that executive information systems (develop) as mainframe computer-based programs. 2. This computer applications (use) so far to satisfy senior executive's needs. 3. I'm sure that great success (achieve) by the company by next term. 4. Computer security problems always (consider) as a significant factor in the development of computer technology. 5. The lecturer explained us that a web server (hide) in a matchbox so that a few people could give an accurate count of the number they had in their homes. 6. By next decade good prevention measures (introduce) to stop unauthorized users from accessing any part of the computer system. 7. Valuable information and services just (protect) from publication by collective processes and mechanisms. 8. It is impossible to determine whether a disclosure or modifications (authorize) properly without authentication. 9. He informed me that information just (share) among companies. 10. I want to remind you that my computer system (secure) recently. 11. He mentioned that the threats for computer security (classify) into several categories. 12. A software flaw (discover) by specialists recently. 13. In two year's time the code from the exploit program (reuse) in Trojan horses. 14. I'm very sorry to say that his private conversation (eavesdrop). 15. I found out that the program (intend) to act as a system of eavesdropping protocols.

Exercise 24. *Use the verbs in brackets in the Active or Passive form of Perfect Tenses.*

1. The plan for construction of the system (create) before we knew about it. 2. I hope the architect (make) a right decision by next meeting. 3. We (not find) any rough mistakes in our research. 4. I found out that each interface in the system (mechanize) with more than one of the coordination protocols. 5. It just (ensure) conceptual integrity. 6. The need to retest components (reduce) by next experiment. 7. Up to now these instructions (reflect) choices about particular analysis. 8. He was sure that it (foster) the creation of the simplest solution to the system problem. 9. Lately all solutions (take into account) and a right one (choose). 10. The software (restore) by next week by a good team of specialists. 11. I think they already (define) their tasks. 12. Their achievements (evaluate) by next summit. 13. Some trade-offs already (find). 14. He

announced that the implementation of a complex functional feature (split) between three groups. 15. Today I (review) multiple software existence planes.

Exercise 25. *Translate into English.*

1. Він повідомив, що кілька складних завдань було виконано цим процесором досить швидко. 2. Ця операційна система буде встановлена до вечора. 3. Нещодавно були розроблені нові версії цієї операційної системи. 4. Вони сказали, що результати їхніх досліджень вже оприлюднені. 5. Всі проблемні питання щодо вдосконалення програмної системи будуть вирішені до наступного місяця. 6. Архітектура програмного забезпечення була презентована досить детально. 7. Я дізнався, що в процес проектування були введені нові важливі компоненти. 8. Деякі критерії покращення процесу планування були щойно розглянуті. 9. Він сказав нам, що всі функціональні вимоги були враховані. 10. Значні переваги цього методу щойно знайшли підтвердження.

Exercise 26. *Study the vocabulary to text 3.*

Compatibility – сумісність
backward-compatible – зворотно сумісний, сумісний «назад» (*такий, що не виключає використання попередніх версій чи модифікацій*)
interoperability – функціональна сумісність, можливість взаємодії *(програмних та апаратних виробів різних постачальників)*
extensibility – розширюваність, можливість розширення (нарощення)
underlying – такий, що лежить в основі, основний, головний, базовий
tolerate – витримувати
fault-tolerance – відмовостійкість
resistant – стійкий
recover – відновлювати(ся)
component failure – відмова елемента
comprise – містити в собі
packaging – пакування
manual – посібник; довідник, покажчик; підручник
enhance – збільшувати, поліпшувати
usability – 1) зручність користування; 2) практичність

reusability – можливість повторного використання
robustness – міцність
invalid – неправильний, недійсний, помилковий,
resilience – стійкість
withstand – протистояти, витримувати
hostile – ворожий
target user – цільовий користувач
target audience – цільова аудиторія
default value – значення за замовчуванням

Exercise 27. *Read and translate text 3.*

Text 3. Design Considerations

There are many aspects to consider in the design of a piece of software. The importance of each should reflect the goals the software is trying to achieve. Some of these aspects are:

- **Compatibility**. The software is able to operate with other products that are designed for interoperability with another product. For example, a piece of software may be backward-compatible with an older version of itself.
- **Extensibility**. New capabilities can be added to the software without major changes to the underlying architecture.
- **Fault-tolerance**. The software is resistant to and able to recover from component failure.
- **Maintainability**. The software can be restored to a specified condition within a specified period of time. For example, antivirus software may include the ability to periodically receive virus definition updates in order to maintain the software's effectiveness.
- **Modularity**. The resulting software comprises well defined, independent components. That leads to better maintainability. The components could be then implemented and tested in isolation before being integrated to form a desired software system. This allows division of work in a software development project.
- **Packaging**. Printed material such as the box and manuals should match the style designated for the target market and should enhance usability. All compatibility information should be visible on the outside of the package. All components required for use should be included in the package or specified as a requirement on the outside of the package.

- **Reliability**. The software is able to perform a required function under stated conditions for a specified period of time.
- **Reusability**. The software is able to add further features and modification with slight or no modification.
- **Robustness**. The software is able to operate under stress or tolerate unpredictable or invalid input. For example, it can be designed with a resilience to low memory conditions.
- **Security**. The software is able to withstand hostile acts and influences.
- **Usability**. The software user interface must be usable for its target user/audience. Default values for the parameters must be chosen so that they are a good choice for the majority of the users.

Exercise 28. *Match the aspects to be considered in the software design with their explanations.*

1. Security	a) The software can be restored to a specified condition within a specified period of time.
2. Reusability	b) New capabilities can be added to the software without major changes to the underlying architecture.
3. Robustness	c) The software is able to add further features and modification with slight or no modification.
4. Usability	d) The software is able to operate with other products that are designed for interoperability with another product.
5. Modularity	e) The software is resistant to and able to recover from component failure.
6. Maintainability	f) The software is able to perform a required function under stated conditions for a specified period of time.
7. Fault tolerance	g) The software is able to withstand hostile acts and influences.
8. Compatibility	h) Printed material such as the box and manuals should match the style designated for the target market and should enhance usability.
9. Packaging	i) The software is able to operate under stress or tolerate unpredictable or invalid input.
10. Extensibility	j) The software user interface must be usable for its target user/audience.

11. Reliability	k) The resulting software comprises well defined, independent components.

Exercise 29. *Answer the questions on text 3.*

1. What aspects should be considered in the design of a piece of software? 2. What is compatibility? 3. How can extensibility be explained? 4. What does fault tolerance mean? 5. What is maintainability? 6. What is modularity and what does it allow? 7. What components and information should be included in the package? 8. What are reliability and reusability? 9. Why is robustness important? 10. What acts is well designed software able to withstand? 11. What is the software user interface designed for?

Exercise 30. *Compose a dialogue on different design aspects.*

Exercise 31. *Study the vocabulary to text 4.*

Enumerate – перелічувати, перераховувати
concern – проблема, питання, справа
black-box – «чорна скринька», пристрій або програма з невідомою внутрішньою структурою
similar – подібним чином
fancy – незвичайний
buggy – що містить значну кількість помилок, *проф.* «глючний»
verify – перевіряти, контролювати
huddle (*together*) – збирати(ся) (*разом*)
multiple – багатократний
open-ended – з можливістю розширення, розширюваний
immaterial – нематеріальний, неістотний
inversion – інверсія, зворотне перетворення
granule – гранула, частинка
release – 1) вивільнення, розблокування; 2) вивільняти(ся)
tracking system – система стеження
closure – закриття, припинення
directed acyclic graph (DAG) – орієнтований ациклічний граф
prove – 1) доводити, засвідчувати, підтверджувати; 2) засвідчувати, підтверджувати документами
proven – доведений, випробуваний, перевірений
paradigm – 1) парадигма; 2) зразок, еталон

pollute – забруднювати
entity – об'єкт
derived – похідний
consistency – логічність, послідовність, зв'язність
skip – пропускати, стрибати, перестрибувати
preceding – попередній
frequently – часто
distil – очищувати
commonality – спільність, уніфікованість
exhaustive – вичерпний, повний, виснажливий
Computer-Aided Software Engineering (CASE) – система автоматизованого розроблення програм, CASE-технологія

Exercise 32. *Read and translate text 4.*

Text 4. Rules of Design

• Make sure that the problem is well-defined.
- All design criteria, requirements, and constraints should be enumerated before a design is started.
- This may require a "spiral model" approach.
• What comes before how.
- i.e., define the service to be performed at every level of abstraction before deciding which structures should be used to realize the services.
• Separate orthogonal concerns.
- Do not connect what is independent.
• Design external functionality before internal functionality.
- First consider the solution as a black-box and decide how it should interact with its environment.
- Then decide how the black-box can be internally organized. Likely it consists of smaller black-boxes that can be refined in a similar fashion.
• Keep it simple.
- Fancy designs are buggier than simple ones; they are harder to implement, harder to verify and often less efficient.
- Problems that appear complex are often just simple problems huddled together.
• Our job as designers is to identify the simpler problems, separate them, and then solve them individually.
• Work at multiple levels of abstraction.

– Good designers must be able to move between various levels of abstraction quickly and easily.
• Design for extensibility.
– A good design is "open-ended," i.e., easily extendible.
– A good design solves a class of problems rather than a single instance.
– Do not introduce what is immaterial.
– Do not restrict what is irrelevant.
– Before implementing a design, build a high-level prototype and verify that the design criteria are met.
• Details should depend upon abstractions.
– Abstractions should not depend upon details.
– Principle of Dependency Inversion.
• The granule of reuse is the same as the granule of release.
– Only components that are released through a tracking system can be effectively reused.
• Classes within a released component should share common closure.
– That is, if one needs to be changed, they all are likely to need to be changed.
– i.e., what affects one, affects all.
• Classes within a released component should be reused together.
– That is, it is impossible to separate the components from each other in order to reuse less than the total.
• The dependency structure for released components must be a DAG.
– There can be no cycles.
• Dependencies between released components must run in the direction of stability.
• The more stable a released component is, the more it must consist of abstract classes.
– A completely stable component should consist of nothing but abstract classes.
• Where possible, use proven patterns to solve design problems.
• When crossing between two different paradigms, build an interface layer that separates the two.
– Don't pollute one side with the paradigm of the other.
• Software entities (classes, modules, etc) should be open for extension, but closed for modification.
– The Open/Closed principle – Bertrand Meyer.

• Derived classes must be usable through the base class interface without the need for the user to know the difference.
– The Liskov Substitution Principle.
• Make it work correctly, then make it work fast.
– Implement the design, measure its performance, and if necessary, optimize it.
• Maintain consistency between representations.
– e.g., check that the final optimized implementation is equivalent to the high-level design that was verified.
• Don't skip the preceding rules!
– Clearly, this is the most frequently violated rule!!!
• Good designs can generally be distilled into a few key principles:
– Separate interface from implementation.
– Determine what is common and what is variable with an interface and an implementation.
– Allow substitution of variable implementations via a common interface.
- i.e., the "open/closed" principle.
– Dividing commonality from variability should be goal-oriented rather than exhaustive.
• Design is not simply the act of drawing a picture using a CASE tool or using graphical UML notation!!!
– Design is a fundamentally creative activity.

Notes

Orthogonal – тут: незалежний

Open/Closed principle – принцип відкриття-закриття *(один з принципів об'єктно-орієнтованого проектування, що дозволяють розробникам усунути помилки проекту, сформувавши найкращий проект на основі наявного набору властивостей)*

Liskov Substitution Principle – принцип підстановки «Лісков» *(запропонований Барбарою Лісков у 1987 році; цей принцип є важливим критерієм оцінювання якості рішень, що ухвалюють під час побудови ієрархій наслідування в об'єктно-орієнтованому проектуванні)*

Exercise 33. *Find in text 4 the English for:*

переконатися, що завдання добре визначене; перераховувати критерії проектування, вимоги та обмеження; розглядати рішення як «чорну скриньку»; деталізувати подібним чином; містити більше

помилок; прості завдання, об'єднані разом; кілька рівнів абстракції; легко розширюваний; відповідати критеріям проектування; принцип інверсії залежностей; орієнтований ациклічний граф; принцип підстановки «Лісков»; оцінювати функціонування; ігнорувати попередні правила; порушуване правило; система автоматизованого розроблення програм.

Exercise 34. *Discuss the rules of design. Which of them are the most important/ more often used/ can be skipped? Can you add any other rules to those listed above?*

Exercise 35. *Prepare a presentation on one of the topics:*

- "Software Design Fundamentals"
- "Software Design Context"
- "Key Issues in Software Design"
- "Concurrency"
- "Design Patterns"
- "Software Design Notations"
- "Software Design Strategies and Methods"
- "Function-Oriented -Structured Design"
- "Object-Oriented Design"
- "Data-Structure-Centered Design"
- "Component-Based Design"

UNIT 7. DATABASES

Exercise 1. *Study the basic vocabulary.*

a) terms
database – база даних
content – зміст, інформаційне наповнення, контент
database management system (DBMS) – система керування базами даних (СКБД)
database architecture – архітектура, конфігурація бази даних
relational database model– реляційна модель бази даних
computing system – обчислювальна система
server cluster – серверний кластер
query language – мова запитів (*мова керування даними*)
Structured Query Language (SQL) – мова структурованих запитів

XQuery– перехресна мова запитів
Extensible Markup Language (XML) – розширювана мова гіпертексту

b) nouns
collection – сукупність, набір
scalability – розширюваність, масштабування
security – безпека
backup – резервне копіювання
facilities – 1) засоби; 2) можливості
entry – елемент, позиція

c) verbs
involve – передбачати, *тут:* бути пов'язаним
manage – керувати
dominate – переважати, панувати, домінувати
operate – керувати, управляти
categorize – класифікувати
cover – охоплювати, містити в собі
implement – реалізовувати

d) adjectives
bibliographic – бібліографічний
document-text – документно-текстовий
operational – експлуатаційний
uncomplicated – неускладнений
specialized – спеціалізований
identical – однаковий, тотожний, ідентичний

e) adverbs
typically – зазвичай
clearly – очевидно, зрозуміло
physically – фізично

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Consist, organized, physically, document, statistical, define, access, external, cover, type, conceptual, implement, architecture, concern, uncomplicated, backup, clearly, extensible, manage, allowance, operate, specialized, security, bibliographic, involve, typically, entry.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) collection, use, typically, involve, type, manage, data, maintenance, search, clearly, separate, major, feature, model, dominate, view, computing, architecture, cost, operational, provide, categorize, support, performance, speed, cover, specific, requirement, identical;

b) digital, allow, internal, clearly, separate, major, understand, single, physically, common, complicated, software, security, relational, mobile, maximum, multiple, similar.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: collect – collection – collector – collective – collectively

Collect, category, maintain, type, classify, use, architect, clear, secure, special, internal, digit, manage, concept, compute, perform, store, external, direction, create, physical, serve, structure, relate, accord, operate, implement.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

An organized collection of data; bibliographic, document-text and statistical type of database content; digital databases; a database management system; to store database contents; to allow data creation and maintenance; search and access; database architecture; external, conceptual and internal levels; a major feature of the relational database model; a single database; to be physically stored and processed by the computing system; internal architecture; to be concerned with cost, performance, scalability and other operational matters; to provide storage, access, security, backup and other facilities; to support relational or XML database model; a server cluster; multiple query languages, such as SQL or XQuery; to access the database; performance trade-offs, such as maximum scale or maximum speed; to cover more than one entry in these categories; offer specific features for more specialized requirements; to be similar, but not identical.

Exercise 6. *Read and translate text 1.*

Text 1. Database Architecture and Management

A database consists of an organized collection of data for one or more uses, typically in digital form. One way of classifying databases

involves the type of their contents, for example: bibliographic, document-text, statistical. Digital databases are managed using database management systems, which store database contents, allowing data creation and maintenance, search and other access.

Database architecture consists of three levels: external, conceptual and internal. Clearly separating the three levels is a major feature of the relational database model that dominates in 21st century databases.

The external level defines how users understand the organization of the data. A single database can have any number of views at the external level. The internal level defines how the data is physically stored and processed by the computing system. Internal architecture is concerned with cost, performance, scalability and other operational matters. The conceptual level provides a common view of the database that is uncomplicated by details of how the data is stored or managed.

A database management system (DBMS) consists of software that operates databases, providing storage, access, security, backup and other facilities. Database management systems can be categorized according to the database model that they support, such as relational or XML, the type of computer they support, such as a server cluster or a mobile phone, the query languages that access the database, such as SQL or XQuery, performance trade-offs, such as maximum scale or maximum speed or others. Some database management systems cover more than one entry in these categories, supporting multiple query languages.

Most DBMSs as of 2009 implement a relational model. Other DBMS systems, such as Object DBMS, offer specific features for more specialized requirements. Their components are similar, but not identical.

Exercise 7. *Find in text 1 the English for:*

організований набір даних; система керування базами даних; інформаційне наповнення бази даних; створення даних; зовнішній, концептуальний та внутрішній рівні; реляційна модель бази даних; обчислювальна система; програмне забезпечення; забезпечувати зберігання, доступ, безпеку, резервне копіювання та інші можливості; серверний кластер; мова структурованих запитів і перехресна мова запитів; компромісне співвідношення функціональних характеристик; максимальне охоплення або максимальна швидкість; розширювана мова гіпертексту.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. A database consists of an organized collection of data for one or more uses, typically in analogue form. 2. One way of classifying databases involves the type of their contents, for example: bibliographic, document-text, statistical. 3. Digital databases are managed using database management systems, which store database contents, allowing data creation and maintenance, and search and other access. 4. Database system consists of three levels, external, conceptual and internal. 5. The internal level defines how users understand the organization of the data. 6. External architecture is concerned with cost, performance, scalability and other operational matters. 7. The conceptual level provides a common view of the database that is uncomplicated by details of how the data is stored or managed. 8. A database management system (DBMS) consists of hardware that operates databases, providing storage, access, security, backup and other facilities. 9. Database management systems can be categorized according to the database model and type(s) of computer that they support, the query language(s) that access the database, performance trade-offs, such as maximum scale or maximum speed or others. 10. Most DBMS as of 2009 offer specific features for more specialized requirements.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to consist of	a) cost, performance, scalability
2) to store	b) according to the database model
3) to allow	c) an organized collection of data
4) to be processed by	d) multiple query languages
5) to be concerned with	e) a relational model
6) to provide	f) database contents
7) to be categorized	g) specific features
8) to support	h) the computing system
9) to implement	i) data creation and maintenance
10) to offer	j) storage, access, security, backup and other facilities

Exercise 10. *Fill in the blanks with prepositions* ***with, in, to, for, at, by, of*** *where necessary.*

1. A database consists … an organized collection … data … one or more uses, typically … digital form. 2. One way … classifying databases involves the type … their contents, … example: bibliographic, document-text, statistical. 3. Clearly separating the three levels is a major feature … the relational database model that dominates … 21st century databases.. 4. A single database can have any number … views … the external level. 5. Internal architecture is concerned … cost, performance, scalability and other operational matters. 6. The conceptual level provides a common view … the database that is uncomplicated … details … how the data is stored or managed. 7. Database management systems can be categorized according … the database model that they support. 8. Other DBMS systems, such as Object DBMS, offer specific features … more specialized requirements. 9. A single database can have any number … views … the external level.

Exercise 11. *Fill in the blanks with proper terms (****database architecture, external level, database management system, internal level, database, conceptual level, object DBMS****) to complete the sentences.*

1._____ is an organized collection of data for one or more uses, typically in digital form. 2. _____consists of software that operates databases, providing storage, access, security, backup and other facilities. 3. _____ is a combination of external, conceptual and internal levels. 4. _____ defines how users understand the organization of the data. 5. _____ is a level of indirection between internal and external. 6. _____ defines how the data is physically stored and processed by the computing system. 7. _____ is a system that offers specific features for more specialized requirements.

Exercise 12. *Answer the questions on text 1.*

1. What does a database consist of? 2. How can databases be classified? 3. What types of database contents are known? 4. What functions do database management systems perform? 5. What levels does database architecture consist of? 6. What do the external and internal levels of database architecture define? 7. What does the conceptual level provide?

8. What does a database management system include? 9. What categories of database management systems can be defined? 10. What features does Object DBMS offer?

Exercise 13. *Put all possible questions to the sentences below.*

1. A database consists of an organized collection of data for one or more uses, typically in digital form. 2. Digital databases are managed using database management systems, which store database contents, allowing data creation and maintenance, and search and other access. 3. Clearly separating three levels was a major feature of the relational database model that dominates in 21st century databases. 4. A single database can have any number of views at the external level. 5. The conceptual is a level of indirection between internal and external. 6. Database management systems can be categorized according to the database model that they support, such as relational or XML. 7. Other DBMS systems, such as Object DBMS, offer specific features for more specialized requirements.

Exercise 14. *Translate into English.*

1. База даних складається з організованого набору даних для одного чи більше застосувань, зазвичай у цифровій формі. 2. Один зі способів класифікування баз даних пов'язаний з типом їх вмісту, наприклад: бібліографічний, документно-текстовий, статистичний. 3. Цифровими базами даних керують з використанням систем керування, які зберігають вміст баз даних, дозволяючи створення даних, пошук та інший доступ. 4. Архітектура бази даних складається з трьох рівнів: зовнішнього, концептуального та внутрішнього. 5. Чітке виокремлення цих трьох рівнів є важливою ознакою реляційної моделі баз даних, яка домінує серед баз даних 21 століття. 6. Зовнішній рівень визначає, як користувачі розуміють організацію даних. 7. Внутрішній рівень визначає, як дані зберігаються фізично і обробляються обчислювальною системою. 8. Внутрішня архітектура стосується вартості, функціонування, розширюваності та інших експлуатаційних питань. 9. Концептуальний рівень дає загальне уявлення про базу даних, тобто не містить подробиць щодо того, як здійснюється зберігання даних і керування ними. 10. Система керування базами даних складається з програмного

забезпечення, яке керує інформаційними базами, забезпечуючи зберігання, доступ, безпеку, резервне копіювання та інші можливості. 11. Системи керування базами даних класифікують відповідно до моделі бази даних та типу комп'ютерів, мов запитів, які мають доступ до цієї бази даних, та можливих компромісне співвідношення функціональних характеристик. 12. Об'єктні системи баз даних пропонують особливі властивості для більш спеціальних потреб. Їх компоненти і подібні, проте не ідентичними.

Exercise 15. *Retell the text* "*Database Architecture and Management"*.

Exercise 16. *Study the vocabulary to text 2.*

Operational database – операційна база даних
subject matter – тематика, зміст
relatively – відносно, порівняно
transaction – транзакція *(самостійне або завершене повідомлення про якусь подію або стан, зафіксоване на якомусь носії інформації і призначене для ініціювання операції СКБД)*
customer database – клієнтська база даних
business – *(комерційна)* організація
personnel database – база даних персоналу, кадрів
benefits – пільги та допомога, соцпакет
skills data – дані про кваліфікацію
manufacturing database – технологічна база даних
inventory – 1) інвентаризація; переоблік; 2) наявні товари; товарно-матеріальні запаси
financial database – база даних для оброблення фінансової інформації
keep track of – відслідковувати
accounting – бухгалтерський облік; фінансова звітність
financial dealings – фінансові оборудки
data warehouse – сховище даних
archive – архівувати, поміщувати до архіву
historical data – ретроспективні дані *(у базах даних – сукупність даних, отриманих за тривалий період, час)*
market research – вивчення кон'юнктури, стану ринку

operational data – 1) робочі дані; 2) дані щодо функціонування системи, експлуатаційні дані
summarize – 1) підсумовувати; 2) зводити, узагальнювати
reclassify – рекласифікувати, змінювати класифікацію, переводити до іншої категорії, перегруповати
analytic(al) database – аналітична база даних
against – *тут:* з
extract – вибирати, отримувати
sales records – торговельна статистика, реєстрація обсягів продажу
advertising – реклама, рекламування
sales promotion – просування товару, стимулювання збуту
aggregate level – агрегований, сукупний, комплексний рівень
distributed database – розподілена база даних
branch office – відділення, філія
manufacturing plant – виробництво, підприємство
work site – робоче місце, об'єкт *(виконання робіт)*
user database – база даних користувачів (абонентів)
end-user database – база даних кінцевих користувачів
download – завантажувати, скачувати
subscription – передоплата, плата наперед
hypermedia database – гіпермедійна база даних
World (WWW) – «Всесвітнє павутиння», глобальна гіпертекстова система Internet
web browser – браузер, програма веб-перегляду, навігатор
while – в той час як, тоді як
web crawler – пошуковий агент, «павук»
database index – індекс (*показник*) бази даних

Exercise 17. *Translate the word combinations below into Ukrainian.*

Operational database; to store detailed data; to process relatively high volumes of updates; customer databases; to record contact, credit, and demographic information about business customers; personnel databases; to hold information such as salary and benefits; skills data about employees; manufacturing databases; to record details about product components and parts inventory; financial databases; keep track of the organization's money, accounting and financial dealings; data warehouse; external sources such as market research firms; to undergo transformation; to get summarized or reclassified.

Exercise 18. *Read and translate text 2.*

Text 2. Types of Databases

Operational database. These databases store detailed data about the operations of an organization. They are typically organized by subject matter, process relatively high volumes of updates using transactions. Essentially every major organization on the earth uses such databases. Examples include customer databases that record contact, credit, and demographic information about business customers; personnel databases that hold information such as salary and benefits; skills data about employees; manufacturing databases that record details about product components, parts inventory, and financial databases that keep track of the organization's money, accounting and financial dealings.

Data warehouse. Data warehouses archive historical data from operational databases and often from external sources such as market research firms. Often operational data undergo transformation on the way into the warehouse, getting summarized, reclassified, etc. The warehouse becomes the central source of data for use by managers and other end-users who may not have access to operational data.

Analytical database. Analysts may do their work directly against a data warehouse, or create a separate analytic database for online analytical processing. For example, a company might extract sales records for analyzing the effectiveness of advertising and other sales promotions at an aggregate level.

Distributed database. These are databases of local work-groups and departments at regional offices, branch offices, manufacturing plants and other work sites. These databases can include segments of both common operational and common user databases, as well as data generated and used only at a user's own site.

End-user database. These databases consist of data developed by individual end-users. Examples of these are collections of documents in spreadsheets, word processing and downloaded files, or even managing their personal card collection.

External database. These databases contain data collection for use across multiple organizations, either freely or via* subscription. The Internet Movie Database is one example.

* via – через, за допомогою, шляхом

Hypermedia databases. The World Wide Web can be thought of as a database, through one spread across millions of independent computing systems. Web browsers "process" this data one page at a time, while web crawlers and other software provide the equivalent of database indexes to support search and other activities.

Exercise 19. *Find in text 2 the English for:*

велика організація; клієнти організації; архівувати ретроспективні дані з операційних баз даних; перелік запчастин; мати доступ до робочих даних; створювати окрему аналітичну базу даних; ефективність реклами та просування товару; розподілена база даних; регіональні офіси, філії, виробничі підприємства та інші об'єкти; набір документів у вигляді електронних таблиць; оброблення текстів; завантажені файли; набір даних для використання багатьма організаціями; шляхом передплати; гіпермедійна база даних; програма веб-перегляду, пошуковий агент та інше програмне забезпечення; підтримувати пошук та інші операції.

Exercise 20. *Answer the questions on text 2.*

1. What does an operational database store? 2. How are operational databases typically organized? 3. What types of operational databases are known? 4. What do data warehouses archive? 5. Who can use the data warehouse as the central source of data? 6. What do analysts create a separate analytic database for? 7. What is referred to as a distributed database? 8. What kind of data do end-user databases consist of? Can you give an example of the end-user database? 9. What does an external database contain? 10. What kind of database is spread across millions of independent computing systems? 11. What can web browsers and web crawlers provide?

Exercise 21. *Make up questions to the italicized parts of the sentences.*

1. ***Operational databases*** store detailed data about the operations of an organization. 2. ***Data warehouses*** archive ***historical data from operational databases and often from external sources such as market research firms***. 3. ***Distributed databases*** can include segments of ***both common operational and common user databases***. 4. ***The end-user*** database consists of ***data developed by individual end-users***. 5. ***External databases*** contain data collection for ***use across multiple***

organizations. 6. ***Web crawlers and other software*** provide ***the equivalent of database indexes*** to support search and other activities.

Exercise 22. *Give nouns corresponding to the following verbs. Translate them.*

Construct, verify, apply, define, reduce, achieve, create, anticipate, communicate, program, operate, automate, restrict, direct, manage, interact, organize, specify, coordinate, act, provide, distribute, store, use, subscribe, develop, generate.

Exercise 23. *Use the proper form of the Perfect or Perfect-Continuous Tense of the verbs in brackets.*

1. They (gather) data since last month. 2. He said that he (search) for any valuable information all day yesterday. 3. When they finish I (wait) for them for thirty minutes. 4. I knew that he (operate) this department for three years. 5. I (analyze) the effectiveness of advertising for two hours when the boss came. 6. Individual end-users (develop) these databases day after day for years. 7. He (enforce) databases security for ten years, why don't you ask him for help? 8. Next week they (protect) their databases from unauthorized activity for a month. 9. He (manage) his project since last month? 10. I wanted to know whether they (try) a new method when you called them? 11. You (work) here for fifteen years by the time you retire? 12. I thought he (work) with unauthorized activity for a long time before. 13. By next semester I (study) this science for four years. 14. For two hours now we (choose) right techniques and we (not come) to any agreement yet. 15. I wondered if they (use) LCD's for a long time. 16. He (repair) peripheral devices for five years on Monday. 17. A new specialist (try) to add a variety of features to his device since early morning. 18. He mentioned that he (implement) interactive computing systems for human use. 19. I (teach) engineering and design methods for two hours when you only enter the university. 20. I asked him if he (work) as a programmer since he graduated from university. 21. He still (improve) his skills for a week? 22. What you (do) with this device? It isn't working! 23. They (try) to eliminate the need for separate power supplies before I came. 24. Since he started his work, he (contribute) to the general success due to his exceptional skills. 25. How long they (investigate) the accident before they left. 26. He (write) his report for three hours when she arrives. 27. I (not repair) my media player yet. 28. They (take) com-

puter-based training course before they started teaching. 29. The users (interact) for hours when he arrived. 30. I wonder what he (do) with this keypad so long.

Exercise 24. *Put the verbs in brackets into the proper Perfect or Perfect Continuous form.*

1. My friend is in his laboratory. He (work) hard on his experiment since early morning. 2. I knew that since that time they (apply) these paradigms in such areas as engineering and telecommunications. 3. The employees (prepare) their reports for two days when the head of the department arrives. 4. I wonder what you (discuss) since two o'clock. 5. "Don't worry, they (work) with such equipment for a long time," he said. 6. He found out that I already (make) an attempt to avoid the overhead. 7. He (head) this department for two years when I began working in it. 8. They (develop) the software for a long period of time. 9. "Look attentively! He (try) to solve this task in his own way." 10. He (look through) his report several times before he realized his mistake. 11. He said that they (produce) this equipment since 2005. 12. In 2012 she (teach) programming at university for ten years. 13. He (install) a software driver since morning? 14. He asked why I (study) the assignment for so many hours.15. I found out that he (work) there for ten years.

Exercise 25. *Use the proper tense and voice form of the verb in brackets.*

1. I'm proud that malicious individuals (not penetrate) yet our well-designed computer system. 2. She said that she (take advantages) of this approach for a long period of time. 3. A serious bug already (find) in the program. 4. He (observe) this malfunction since morning. 5. The lecturer explained us that the denial of service attacks (not use) to gain unauthorized access or control of a system. 6. I'm sure all types of attacks (prevent) successfully by next month owing to his experience. 7. They told us that they already (analyze) all possible human errors. 8. By next year he (work) in this department for ten years. 9. I must admit that I (not find) any deception in his activity. 10. What kind of problem (find) lately? 11. I (find) some vulnerabilities very quickly. 12. By tomorrow they (work) on this project for a month. 13. They (download) large quantities of data onto backup media since yesterday. 14. I never (encrypt) the storage media. 15. Some unsuccessful attempts (make) before he came. 16. This kind of software (design) before he knew about it.

17. You (achieve) your goals by next semester? 18. In three years' time he (accomplish) this work. 19. Some malicious programs just (detect) by the administrator. 20. By nine o'clock he already (install) the program. 21. We (try) to make the computer work since five o'clock. 22. He said that the infection (transmit) by e-mail or Microsoft word documents. 23. I think that you (not get) the consent before I come. 24. The malicious software (infect) the worm into user's local networks for years. 25. It was clear that he (try) to avoid detection and disinfection for a long time.

Exercise 26. *Change the sentences into indirect speech.*

1. I asked my teacher, "How long have you been teaching this subject?" 2. The manager said, "I have just looked through your personal data." 3. My friend told me, "I have been trying to enforce the security of my system through access control." 4. They asked me, "Where did you get the experience in this work?" 5. She told them, "I have been studying object-oriented programming for two years." 6. He asked, "Did you give a right definition to this term last time?" 7. We wondered, "Who managed to solve these difficult problems?" 8. The teacher asked me, "Did you complete your assignment?" 9. The dean said, "You have already proved your serious attitude to studies." 10. My colleague exclaimed, "This is the best job I have ever dreamt of!"

Exercise 27. *Translate the sentences into English. Mind the sequence of tenses.*

1. Ми знаємо, що він займається модернізуванням програмного забезпечення вже впродовж багатьох років. 2. Я думав, що шкідливі програми виявлені. 3. Вона каже, що обов'язково використає резервні носії до кінця тижня. 4. Ми не знаємо, чи вони вже отримали цю порцію даних. 5. Цікаво, чому виявляють ці файли так довго. 6. Я помітив, що він вже запускав раніше такі програми. 7. Я впевнена, що мій колега виявить це порушення до вечора. 8. Ми бачили, що вона знайшла потрібний метод вирішення складної задачі. 9. Викладач пояснив, що він вже раніше успішно використовував такі методики на практиці. 10. Він повідомив, що наступного року виповниться 20 років, як він працює системним аналітиком. 11. Я була впевнена, що вони використовують розширювану мову гіпертексту вже довгий час. 12. Вони запитали, коли я отримав доступ

до цієї бази даних. 13. Нам вчора повідомили, що такі випробування проводились раніше досить часто. 14. Він обіцяє, що вивчить цю обчислювальну систему до кінця тижня. 15. Ми дізналися, що наші колеги підтримували реалізацію його сумнівного проекту. 16. Я маю сумніви, що вона вже ознайомилась з конфігурацією бази даних. 17. Я хочу знати, що вони вже спланували на наступний навчальний рік. 18. Він здивувався, що ми так довго вивчали новий матеріал. 19. Вона запитала мене, де я знайшов інформацію про цю систему. 20. Він впевнений, що ми отримаємо пільги та допомогу до наступного місяця.

Exercise 28. *Study the vocabulary to text 3.*

Denote – означати
unauthorized activity – неправомірна, несанкціонована діяльність, несанкціоновані дії
enforce – зміцнювати, посилювати
auditing – ревізія / перевірка системи, *сленг.* аудитування
encryption – кодування, шифрування
authentication – автентифікація, підтвердження автентичності, ідентифікація
authorization – авторизація
transmit – передавати,
encrypt – шифрувати, кодувати
decrypt – розшифровувати
query result – результат запиту
confidentiality – конфіденційність, секретність
govern – керувати
release – 1) публікація; 2) надання
driving record – особова картка водія
log – журнал реєстрації
Office – управління уповноваженого з питань інформації
lock (out) – блокувати, відмикати
shared lock – блокування із забезпеченням спільного доступу *(до набору даних)*
exclusive lock – блокування із забезпеченням взаємовиключального доступу *(до набору даних)*
read lock – блокування зчитування

write lock – блокування запису
deadlock – взаємне блокування, *проф.* клінч
ensure – гарантувати, забезпечувати
granularity – рівень модульності *(системи)*, «гранулярність»
coarse – великомодульний
fine-grained – дрібномодульний
intermediate – проміжний
Relational Database Management System (RDBMS) – система керування реляційною базою даних
implicitly – неочевидно, імпліцитно, всередині
explicitly – очевидно, експіцитно, експліковано, зовні
persist – залишатися, зберігатися, продовжувати існувати
isolation – ізольованість
reduce – зменшувати, знижувати, послаблювати
concurrency – 1) паралелізм; 2) взаємна сумісність *(властивість об'єктів в об'єктно-орієнтованому програмуванні)*
abort – переривати, аварійно завершувати

Exercise 29. *Translate the word combinations below into Ukrainian.*

Database security; to denote the system, processes, and procedures; to connect to the database via authentication; to record information about database activity; to encrypt and decrypt data; to protect medical history, driving records, telephone logs; the Office of the Information Commissioner; to hold personal data in digital format such as databases; locking; to modify a resource; to provide one method of ensuring data; granularity; to cover an entire database; rows in a RDBMS table; shared or exclusive locks; to be created implicitly by the DBMS; explicitly at the transaction's request; to read and write locks; to reduce performance and/or concurrency.

Exercise 30. *Read and translate text 3.*

Text 3. Database Security

Database security denotes the system, processes, and procedures that protect a database from unauthorized activity. DBMSs usually enforce security through access control, auditing, and encryption.

Access control manages who can connect to the database via authentication and what they can do via authorization. *Auditing* records information about database activity: who, what, when, and possibly

where. *Encryption* protects data at the lowest possible level by storing and possibly transmitting data in an unreadable form. The DBMS encrypts data when it is added to the database and decrypts it when returning query results. This process can occur on the client side of a network connection to prevent unauthorized access at the point of use.

Confidentiality. Law and regulation governs the release of information from some databases, protecting medical history, driving records, telephone logs, etc. Organizations based in the United Kingdom and holding personal data in digital format such as databases must register with the Office of the Information Commissioner.

Locking. When a transaction modifies a resource, the DBMS stops other transactions from also modifying it, typically by locking it. Locks also provide one method of ensuring that data does not change while a transaction is reading it or even that it doesn't change until a transaction that once read it has completed.

Granularity. Locks can be coarse, covering an entire database, fine-grained, covering a single data item, or intermediate covering a collection of data such as all the rows in a RDBMS table.

Lock types. Locks can be shared or exclusive, and can lock out readers and/or writers. Locks can be created implicitly by the DBMS when a transaction performs an operation, or explicitly at the transaction request. Shared locks allow multiple transactions to lock the same resource. The lock persists until all such transactions complete. Exclusive locks are held by a single transaction and prevent other transactions from locking the same resource. Read locks are usually shared, and prevent other transactions from modifying the resource. Write locks are exclusive, and prevent other transactions from modifying the resource. On some systems, write locks also prevent other transactions from reading the resource.

Isolation. Isolation refers to the ability of one transaction to see the results of other transactions. Greater isolation typically reduces performance and/or concurrency, leading DBMSs to provide administrative options to reduce isolation.

Deadlocks. Deadlocks occur when two transactions each require data that the other has already locked exclusively. Deadlock detection is performed by the DBMS, which then aborts one of the transactions and allows the other to complete.

Exercise 31. *Find in text 3 the English for:*

захищати базу даних від несанкціонованих дій; система керування базою даних; посилювати безпеку через керування доступом, ревізію системи та кодування; зберігання і передавання даних в нечитальній формі; результати запитів; з клієнтського боку мережного з'єднання; оприлюднення інформації; дозволяти багаторазові транзакції; не дозволяти іншим транзакціям змінювати ресурс; вживати адміністративних заходів; виявлення взаємоблокування; переривати одну з транзакцій.

Exercise 32. *Put all possible questions to the sentences below.*

1. Database security denotes the system, processes, and procedures that protect a database from unauthorized activity. 2. DBMSs usually enforce security through access control, auditing, and encryption. 3. Organizations based in the United Kingdom and holding personal data in digital format such as databases must register with the Office. 4. Exclusive locks are held by a single transaction and prevent other transactions from locking the same resource. 5. Deadlock detection is performed by the DBMS, which then aborts one of the transactions and allows the other to complete.

Exercise 33. *Give nouns corresponding to the following verbs. Translate them.*

Denote, protect, enforce, connect, store, transmit, encrypt, decrypt, add, govern, register, modify, provide, ensure, cover, create, perform, allow, persist, prevent, refer, reduce, lead, analyze, justify, require, abort.

Exercise 34. *Find the opposites of the following words in text 3 and using them make up sentences of your own.*

Authorized, weaken, ensure, decrypt, openness, analogue, coarse, exclusive, implicitly, single, different, readable, continue, start, increase.

Exercise 35. *Answer the questions on text 3.*

1. What does database security denote? 2. What can access control manage? 3. What kind of information does auditing record? 4. How is data protected by means of encryption? 5. What do law and regulation govern? 6. How does the DBMS stop other transactions from modifying? 7. What types of locks are known and what are they

designed for? 8. What can locking significantly affect? 9. What does isolation refer to and what does it typically reduce? 10. When do deadlocks usually occur?

Exercise 36. *Find some additional information and speak on database architecture and management systems.*

Exercise 37. *Study the vocabulary to text 4.*

Post-relational database model – пост-реляційна модель бази даних
alternate – альтернативний, інший
hybrid database – гібридна база даних, база даних зі змішаною («*гібридною*») структурою
object-enhanced – об'єктно-розширений
incorporate – об'єднувати, містити у своєму складі
cast (*p., pp.* cast) – *тут:* представляти
in terms of – через, у вигляді
pre-date – передувати, відбуватися, статися (*перед чимось*)
plausible – переконливий, виправданий
object database model – модель об'єктної бази даних
object oriented paradigm – об'єктно-орієнтована парадигма
object-oriented programming – об'єктно-орієнтоване програмування
engineering database – конструкторська база даних
spatial database – розосереджена база даних
conglomeration – конгломерат, суміш
overhead – 1) службові, протокольні сигнали або дані; 2) витрати обчислювальних ресурсів; 3) накладні витрати
impedance mismatch – розузгодженість інтерфейсів *(розбіжність між інтерфейсами об'єкта бази даних)*
variety – 1) різноманітність; 2) відмінність, розбіжність

Exercise 38. *Translate the word combinations below into Ukrainian.*

Relational and post-relational database models; hybrid database; object-enhanced relational database model system (RDBMS); to extend relational systems with non-relational features; historically pre-relational products; the application-programming world; to avoid the overhead of converting information; to convert information between its representation in the database and in the application program; to

introduce key ideas of object programming; to approach the problem from the application-programming side; to make the objects persistent; to require some kind of query language; to provide language-level functionality; objects based on their information content; to define an object-oriented data model.

Exercise 39. *Read and translate text 4.*

Text 4. Database Models

Post-relational database models. Products offering a more general data model than the relational model are sometimes classified as post-relational. Alternate terms include "hybrid database", "object-enhanced RDBMS" and others. The data model in such products incorporates relations but is not constrained by E.F. Codd's Information Principle, which requires that all information in the database must be cast explicitly in terms of values in relations and in no other way. Some of these extensions to the relational model integrate concepts from technologies that pre-date the relational model. For example, they allow representation of a directed graph with trees on the nodes.

Some post-relational products extend relational systems with non-relational features. Others arrived in much the same place by adding relational features to pre-relational systems. Paradoxically, this allows products that are historically pre-relational to make a plausible claim to be post-relational.

Object database models. In recent years, the object-oriented paradigm has been applied in areas such as engineering and spatial databases, telecommunications and in various scientific domains. The conglomeration of object-oriented programming and database technology led to this new kind of database. These databases attempt to bring the database world and the application-programming world closer together, in particular by ensuring that the database uses the same type system as the application program. This aims to avoid the overhead (sometimes referred to as the impedance mismatch) of converting information between its representation in the database (for example as rows in tables) and its representation in the application program (typically as objects). At the same time, object databases attempt to introduce key ideas of object programming, such as encapsulation and polymorphism, into the world of databases.

A variety of these ways have been tried for storing objects in a database. Some products have approached the problem from the application-programming side, by making the objects manipulated by the program persistent. This also typically requires the addition of some kind of query language, since conventional programming languages do not provide language-level functionality for finding objects based on their information content. Others have attacked the problem from the database end, by defining an object-oriented data model for the database, and defining a database programming language that allows full programming capabilities as well as traditional query facilities.

Exercise 40. *Find in text 4 the English for:*

містити в своєму складі співвідношення; бути поданими у вигляді значень у співвідношеннях; включати в себе ідеї; технології, які передують реляційній моделі; дозволяти представлення; спрямований граф з деревами у вузлах; виправдано претендувати; моделі об'єктних баз даних; конструкторські та розосереджені бази даних; різноманітні наукові сфери; суміш об'єктно-орієнтованого програмування і технології баз даних; інкапсуляція і поліморфізм; зберігання об'єктів у базі даних; звичайні мови програмування; мова програмування баз даних; усі можливості програмування, а також традиційні засоби організації запитів.

Exercise 41. *Answer the questions on text 4.*

1. What products are classified as post-relational? 2 What does the data model in post-relational products incorporate? 3. What concepts do extensions to the relational model integrate? 4. What do post-relational products extend relational systems with? 5. In what areas has object-oriented paradigm been recently applied? 6. What led to the object database model? 7. What do object databases attempt to do? 8. What has been tried for storing objects in a database? 9. Why is addition of some kind of query language required? 10. What does defining a database programming language allow? 11. What are the main features of post-relational and object database models?

Exercise 42. *Make up questions to the italicized parts of the sentences.*

1. Products ***offering a more general data model than the relational model*** are sometimes classified as ***post-relational***. 2. ***Some of these***

extensions to the relational model integrate concepts from technologies ***that pre-date the relational model***. 3. In recent years, ***the object-oriented paradigm*** has been applied ***in areas such as engineering and spatial databases, telecommunications and in various scientific domains***. 4. ***The conglomeration of object-oriented programming and database technology*** led to this ***new kind of database***. 5. ***A variety of these ways*** have been tried ***for storing objects in a database***. 6. ***Conventional programming languages*** do not provide ***language-level functionality*** for finding objects ***based on their information content***.

Exercise 43. *Write derivatives of the words below and explain their meanings.*

Classify, incorporate, require, integrate, extend, arrive, add, apply, lead, use, refer, ensure, avoid, convert, introduce, store, manipulate, provide, base, define.

Exercise 44. *Decipher the abbreviations below:*

DBMS, RDBMS, WWW, SQZ, XMZ, DAG, CASE.

Exercise 45. *Write a summary of the text "Database Models".*

UNIT 8. HUMAN-MACHINE INTERFACE

Exercise 1. *Study the basic vocabulary.*

a) terms

Human-Machine Interface (HMI) – інтерфейс «людина – машина», людино-машинний інтерфейс
operator interface terminal – термінал операторського інтерфейсу, модуль зв'язку оператора з об'єктом
programmable function key – програмована функціональна клавіша
full keypad – повна допоміжна клавіатура
alphanumeric display – цифрово-, цифро-літерний дисплей
graphic display – графічний дисплей
backlit display – екран з підсвічуванням
operating temperature – робоча температура
operating humidity – робоча вологість
flat panel display (FPD) – плоскопанельний дисплей
liquid-crystal display (LCD) – рідкокристалічний дисплей / РК-дисплей
gas plasma technology – газорозрядна технологія

liquid crystal solution – рідкокристалічний розчин
polarizing material – поляризаційний матеріал
passive matrix display – РК-екран / дисплей з пасивною матрицею
active matrix display – РК-екран / дисплей з активною матрицею
gas plasma display – газорозрядний дисплей, плазмовий дисплей
response time – час відгуку *(час, потрібний комп'ютеру для відповіді на запит)*
array of pixels – масив елементів зображення
performance specifications – 1) експлуатаційні / робочі характеристики; 2) вимоги до характеристик
hard drive capacity – обсяг жорсткого диску
input/output (I/O) interface – інтерфейс введення-виведення
I/O parallel peripheral bus – паралельна периферійна шина введення-виведення
I/O port – порт введення–виведення
Ethernet – мережа (протокол, стандарт, технологія) Ethernet
small computer systems interface (SCSI) – інтерфейс малих обчислювальних систем, інтерфейс SCSI
universal serial bus (USB) – універсальна послідовна шина, шина USB
local area network (LAN) – локальна (обчислювальна) мережа, ЛОМ
bus topology – шинна топологія *(топологія ЛОМ, у якій усі абоненти лінійно під'єднані до однієї магістралі (шини) пересилання даних)*
star topology – топологія «зірка» *(топологія ЛОМ, у якій пристрої і комп'ютери з'єднані радіальними лініями з центральним вузлом)*
data transfer rate – швидкість пересилання даних
balanced serial interface – зрівноважений послідовний інтерфейс
device-independent protocol – апаратно-незалежний протокол
software driver – програмний драйвер
real-time clock – годинник реального часу
Power-over-Ethernet (PoE) equipment – устаткування / обладнання живлення через Ethernet
electromagnetic interference (EMI) – електромагнітні завади (*наведення*)
radio frequency interference (RFI) – радіозавади, радіочастотні наведення

National Electronics Manufacturers Association (NEMA) – Національна асоціація виробників електроустаткування, асоціація NEMA

b) nouns

device – пристрій, механізм
knob – куляста ручка, кнопка
lever – важіль керування, регулювання, ручка, руків'я
dimensions – розміри
rating – 1) параметр, розрахункова величина; 2) потужність, номінальна характеристика
sheet – лист (тут: поляризаційного матеріалу)
grid – решітка, сітка
wire – дріт, провід
intersection – перетин, точка, лінія перетину
subpixel – субелемент зображення, субпіксель
stylus – стилус *(перо для введення даних у планшетних ноутбуках і кишенькових ПК)*
shielding – екранування *(спосіб захисту передавального середовища від електромагнітних завад)*
enclosure – 1) корпус; 2) огорожа, загорожа; 3) обгороджування; 4) вкладення (*вміст конверта*)

c) verbs

trap – вміщувати
align – орієнтувати
switch on/off – вмикати/вимикати
react with – вступати в реакцію (*з чимось*)
design – 1) проектувати; 2) призначати

d) adjectives

monochrome – монохромний, одноколірний, однотонний
web-enabled – веб-орієнтований
networkable – такий, що під'єднується до комп'ютерної мережі
harsh – жорсткий, суворий

e)adverbs

directly – безпосередньо, прямо, точно
altogether – зовсім, цілком, цілковито

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Intersection, select, networkable, network, align, device-independent, topology, contain, electromagnetic, environment, meet, universal, rating, react, commonly, feature, appropriate, eliminate, array, directly, altogether, capacity, monochrome, interact, enclosure, intelligent, differ.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) device, include, available, graphic, display, standard, message, control, react, differ, performance, transmission, allow, variety, eliminate, separate, offer, commonly, design, appropriate, harsh, association;

b) improve, programmable, digital, software, important, available, appropriate, device-independent, horizontal, connect, common, peripheral, active, local, balance.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: interact – interaction – interactor – interactive – interactively

Interact, control, provide, process, human, use, current, contain, color, perform, network, connect, power, separate, supply, design, differ, appropriate, intelligent, react, eliminate, allow.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

Human-machine interface; operator interface terminal; to include knobs, levers, and controls; alphanumeric or graphic displays; to use liquid crystal display (LCD) or gas plasma technologies; performance specifications; processor type; random access memory (RAM) and hard drive capacity; to allow connections to peripherals such as mice, keyboards, and modems; small computer system interface (SCSI); universal serial bus (USB); device-independent protocol; low-to-medium speed peripheral device connections; web-enabled or networkable devices; to provide real-time clock support; electromagnetic interference (EMI); to meet standards from the National Electronics Manufacturers Association (NEMA).

Exercise 6. *Read and translate text 1.*

Text 1. Human-Machine Interface Design

Human-machine interfaces (HMI) are operator interface terminals with which users interact in order to control other devices. Some human-machine interfaces include knobs, levers, and controls. Others provide programmable function keys or a full keypad. Devices that include a processor or interface to personal computers (PCs) are also available. Many human-machine interfaces include alphanumeric or graphic displays. For ease of use, these displays are often backlit or use standard messages. When selecting human-machine interfaces, important considerations include devices supported and devices controlled. Device dimensions, operating temperature, operating humidity, and vibration and shock ratings are other important factors.

Many human-machine interfaces include flat panel displays (FPDs) that use liquid crystal display (LCD) or gas plasma technologies. In LCDs, an electric current passes through a liquid crystal solution that is trapped between two sheets of polarizing material. The crystals align themselves so that light cannot pass, producing an image on the screen. LCDs can be monochrome or colour. Colour displays can use a passive matrix or an active matrix. Passive matrix displays contain a grid of horizontal and vertical wires with an LCD element at each intersection. In active matrix displays, each pixel has a transistor that is switched directly on or off, improving response times. Unlike LCDs, gas plasma displays consist of an array of pixels, each of which contains red, blue, and green subpixels. In the plasma state, gas reacts with the subpixels to display the appropriate colour.

Human-machine interfaces differ in terms of performance specifications and I/O ports. Performance specifications include processor type, random access memory (RAM) and hard drive capacity, and other drive options. I/O interfaces allow connections to peripherals such as mice, keyboards, and modems. Common I/O interfaces include Ethernet, Fast Ethernet, RS232, RS422, RS485, small computer systems interface (SCSI), and universal serial bus (USB). Ethernet is a local area network (LAN) protocol that uses a bus or star topology and supports data transfer rates of 10 Mbps. Fast Ethernet is a 100 Mbps specification. RS232, RS422, and RS485 are balanced serial interfaces for the transmission of digital data. Small computer systems interface (SCSI) is an intelligent I/O parallel peripheral bus with a standard,

device-independent protocol that allows many peripheral devices to be connected to the SCSI port. Universal serial bus (USB) is a 4-wire, 12-Mbps serial bus for low-to-medium speed peripheral device connections.

Human-machine interfaces are available with a variety of features. For example, some devices are web-enabled or networkable. Others include software drivers, a stylus, and support for a keyboard, mouse, and printer. Devices that provide real-time clock support use a special battery and are not connected to the power supply. Power-over-Ethernet (PoE) equipment eliminates the need for separate power supplies altogether. Human-machine interfaces that offer shielding against electromagnetic interference (EMI) and radio frequency interference (RFI) are commonly available. Devices that are designed for harsh environments include enclosures that meet standards from the National Electronics Manufacturers Association (NEMA).

Exercise 7. *Find in text 1 the English for:*

людино-машинний інтерфейс; керувати іншими пристроями; програмовані функціональні клавіші та повна допоміжна клавіатура; вібраційні та ударостійкі характеристики; плоскопанельний дисплей; електричний струм; створювати зображення на екрані; монохромні і кольорові дисплеї; використовувати активну та пасивну матрицю; прискорювати час відповіді; плазмовий дисплей; складатися з масиву елементів зображення; інтерфейс введення-виведення; протокол локальної обчислювальної мережі; шинна топологія і топологія «зірка»; швидкість пересилання даних; стандартний апаратно-незалежний протокол; периферійний пристрій; різноманітні властивості; джерело живлення; радіозавади.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. Human-machine interfaces are operator interface terminals with which users interact in order to control other devices. 2. Some personal computers include knobs, levers, and controls. Others provide programmable function keys or a full keypad. 3. Device dimensions, operating mode, operating humidity, and vibration and shock ratings are other important factors. 4. In LCDs, an electric current passes through a liquid crystal solution that is trapped between two sheets of polarizing

material. 5. Black-and-white displays can use a passive matrix or an active matrix. 6. Active matrix displays contain a grid of horizontal and vertical wires with an LCD element at each intersection. 7. In passive matrix displays, each pixel has a transistor that is switched directly on or off, improving response times. 8. Human-machine interfaces differ in terms of performance specifications and I/O ports. 9. Performance specifications include processor type, read-only memory, and hard drive capacity, and other drive options. 10. Ethernet is a wide area network protocol that uses a bus or star topology and supports data transfer rates of 10 Mbps. 11. Small computer systems interface (SCSI) is an intelligent I/O parallel peripheral bus with a standard, device-independent protocol that allows many peripheral devices to be connected to the SCSI port. 12. Universal serial bus is a 4-wire, 12-Mbps serial bus for low-to-medium speed peripheral device connections. 13. Devices that provide real-time clock support use some special equipment and are connected to the power supply. 14. Power-over-Ethernet equipment eliminates the need for alternate power supplies altogether.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to interact	a) response times
2) to include	b) connections to peripherals
3) to use	c) an array of pixels
4) to pass through	d) the need for separate power supplies
5) to improve	e) gas plasma technologies
6) to consist of	f) in order to control other devices
7) to display	g) a liquid crystal solution
8) to differ	h) the appropriate colour
9) to allow	i) knobs, levers, and controls
10) to eliminate	j) in terms of performance specifications and I/O ports

Exercise 10. *Fill in the blanks with prepositions* ***in, on, of, at, with, to, between, through, for, from, as, off*** *where necessary.*

1. Human machine interfaces are operator interface terminals … which users interact … order ... control other devices. 2. … LCDs, an electric current passes … a liquid crystal solution that is trapped … two sheets … polarizing material. 3. Passive matrix displays contain a grid … horizontal and vertical wires … an LCD element … each intersection.

4. … active matrix displays, each pixel has a transistor that is switched directly … or … , improving response times. 5. Unlike LCDs, gas plasma displays consist … an array ... pixels, each ... which contains red, blue, and green subpixels. 6. I/O interfaces allow connections … peripherals such … mice, keyboards, and modems. 7. Human-machine interfaces are available … a variety … features. 8. Devices that are designed … harsh environments include enclosures that meet standards … the National Electronics Manufacturers Association (NEMA).

Exercise 11. *Fill in the blanks with proper terms(**power-over-Ethernet equipment, flat panel displays, LCDs, Ethernet, human machine interfaces, performance specifications, gas plasma displays, universal serial bus (USB), small computer systems interface (SCSI), I/O interfaces**) to complete the sentences.*

1. _____ are operator interface terminals with which users interact in order to control other devices. 2. _____use liquid crystal display (LCD) or gas plasma technologies. 3. _____ can be monochrome or colour. 4. _____ consist of an array of pixels, each of which contains red, blue, and green subpixels. 5. _____ include processor type, random access memory (RAM), and hard drive capacity, and other drive options. 6. _____ allow connections to peripherals such as mice, keyboards, and modems. 7. _____ is a local area network (LAN) protocol that uses a bus or star topology and supports data transfer rates of 10 Mbps. 8. _____ is an intelligent I/O parallel peripheral bus with a standard, device-independent protocol that allows many peripheral devices to be connected to the SCSI port. 9. _____ is a 4-wire, 12-Mbps serial bus for low-to-medium speed peripheral device connections. 10. _____ eliminates the need for separate power supplies altogether.

Exercise 12. *Answer the questions on text 1.*

1. What is a human-machine interface? 2. What do human-machine interfaces include? 3. What do important considerations include when selecting human-machine interfaces? 4. What technologies do flat panel displays employ? 5. What does an electric current pass through in LCDs? 6. What types of LCDs are there? 7. What kinds of matrix do colour displays use? 8. What do passive matrix displays contain? 9. How does the active matrix display work? 10. What do gas plasma displays consist of? How do they produce appropriate colours? 11. What do

human-machine interfaces differ in? 12. What do performance specifications include? 13. What is Ethernet? 14. What is the small computer system interface? What does it allow? 15. What is the universal series bus? 16. What features are human-machine interfaces available with? 17. What do devices providing real-time clock support use? 18. What eliminates the need for separate power supplies in human-machine interfaces? 19. What do human-machine interfaces offer to fight electromagnetic and radio frequency interference? 20. What do devices designed for harsh environments include?

Exercise 13. *Put all possible questions to the sentences below.*

1. Human-machine interfaces are operator interface terminals with which users interact in order to control other devices. 2. In LCDs, an electric current passes through a liquid crystal solution that is trapped between two sheets of polarizing material. 3. Colour displays can use a passive matrix or an active matrix. 4. Performance specifications include processor type, random access memory and hard drive capacity, and other drive options. 5. Ethernet is a LAN protocol that uses a bus or star topology and supports data transfer rates of 10 Mbps. 6. Devices that are designed for harsh environments include enclosures that meet standards from the National Electronics Manufacturers' Association (NEMA).

Exercise 14. *Translate into English.*

1. Людино-машинні інтерфейси – це термінали операторського інтерфейсу, з якими користувачі взаємодіють для керування іншими пристроями. 2. Деякі людино-машинні інтерфейси мають кулясті ручки, важелі та органи керування. Інші мають програмовані функціональні клавіші або повну допоміжну клавіатуру. 3. Багато людино-машинних інтерфейсів мають цифро-літерні та графічні дисплеї із заднім підсвічуванням і використовують стандартні повідомлення. 4. Плоскопанельні дисплеї використовують технології рідкокристалічного екрану та плазмові технології. 5. В рідкокристалічних дисплеях електричний струм проходить крізь рідкокристалічний розчин, який знаходиться між двома листами поляризаційного матеріалу. 6. Рідкокристалічні дисплеї можуть бути монохромними та кольоровими. В кольорових дисплеях використовують пасивну або активну матрицю. 7. Рідкокристалічні дисплеї з пасивною матрицею містять у собі решітку з горизонтальними і вертикальними проводами та елементом РК-дисплею в кожній точці їх

перетину. 8. У рідкокристалічних дисплеях з активною матрицею кожний піксель має транзистор, який безпосередньо вмикається й вимикається, що покращує час відгуку. 9. На відміну від РК-дисплеїв, плазмові дисплеї складаються з масиву елементів зображення, кожний з яких містить червоний, синій та зелений субпікселі. 10. У плазмовому стані газ вступає в реакцію з цими субпікселями для відображення потрібного кольору. 11. Людино-машинні інтерфейси відрізняються робочими характеристиками і портами введення-виведення. 12. Робочі характеристики включають тип процесора, оперативну пам'ять, обсяг оперативної пам'яті та жорсткого диску та інші опції. 13. Мережа Ethernet – це протокол локальної обчислювальної мережі, який використовує шинну топологію і топологію «зірка» та підтримує швидкість пересилання даних на рівні 10 мегабіт на секунду. 14. Інтерфейс малих обчислювальних систем – це паралельна периферійна шина введення-виведення зі стандартним апаратно-незалежним протоколом, що дозволяє з'єднувати багато периферійних пристроїв з портом SCSI. 15. У наявності є людино-машинні інтерфейси, які забезпечують екранування від електромагнітних та радіозавад.

Exercise 15. *Retell the text "Human Machine Interface Design".*

Exercise 16. *Study the vocabulary to text 2.*

Human–computer interaction (HCI) – взаємодія людини з машиною
be regarded as – розглядатися як
behavioural sciences – біхевіоризм
large-scale computerized system – велика обчислювальна система
power plant – 1) силова установка; 2) електростанція
Association for Computing Machinery – Асоціація з обчислювальної техніки
surround – оточувати
facet – аспект, сторона
draw (*p*. drew, *pp*. drawn) (*from*) – добувати, діставати, брати (*з чогось*)
development environment – середовище розроблення
cognitive psychology – когнітивна психологія
multidisciplinary – політематичний; багатопрофільний
background – 1) походження, біографічні дані людини; 2) кваліфікація, освіта, *(фахова)* підготовка
contribute – сприяти

investigation – розслідування
conclude – робити висновок
responsible – відповідальний

Exercise 17. *Translate the word combinations below into Ukrainian.*

Human–computer interaction (HCI); interaction between people (users) and computers; the intersection of computer science, behavioural sciences, design and several other fields of study; user interface; to include both software and hardware; input received from users via hardware peripherals; large-scale computerized systems; Association for Computing Machinery; major phenomena; design, evaluation and implementation of interactive computing systems; techniques in computer graphics, operating systems, programming languages, and development environments; cognitive psychology; due to the multidisciplinary nature of HCI; man–machine interaction (MMI); computer–human interaction (CHI); be responsible for the disaster.

Exercise 18. *Read and translate text 2.*

Text 2. Human–Computer Interaction

Human–computer interaction (HCI) is the study of interaction between people (users) and computers. It is often regarded as the intersection of computer science, behavioural sciences, design and several other fields of study. Interaction between users and computers occurs at the user interface, which includes both software and hardware; for example, characters or objects displayed by software on a personal computer‘s monitor, input received from users via hardware peripherals such as keyboards and mice, and other user interactions with large-scale computerized systems such as aircraft and power plants. The Association for Computing Machinery defines human-computer interaction as “a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.” An important facet of HCI is the securing of computer user satisfaction.

Because human-computer interaction studies a human and a machine in conjunction, it draws from supporting knowledge on both the machine and the human side. On the machine side, techniques in

computer graphics, operating systems, programming languages, and development environments are relevant. On the human side, communication theory, graphic and industrial design disciplines, linguistics, social sciences, cognitive psychology, and human factors are relevant. Engineering and design methods are also relevant. Due to the multidisciplinary nature of HCI, people with different backgrounds contribute to its success. HCI is also sometimes referred to as man–machine interaction (MMI) or computer–human interaction (CHI).

Attention to human-machine interaction is important, because poorly designed human-machine interfaces can lead to many unexpected problems. A classic example of this is the Three Mile Island* accident where investigations concluded that the design of the human-machine interface was at least partially responsible for the disaster.

Exercise 19. *Find in text 2 the English for:*

символи або об'єкти, що відображаються програмним забезпеченням на моніторі персонального комп'ютера; важливий аспект взаємодії людини з машиною; забезпечення задоволеності користувача; теорія передавання інформації; графічний та промисловий дизайн; інженерні та конструкторські методи; різні за походженням і освітою люди; неякісно розроблений; призводити до багатьох несподіваних проблем.

Exercise 20. *Answer the questions on text 2.*

1. What is called human-computer interaction? 2. Where does interaction between users and computers occur? 3. What does the user interface include? 4. How does the Association for Computing Machinery define human-computer interaction? 5. What is an important facet of HCI? 6. What does human-computer interaction draw supporting knowledge from? 7. What is HCI also sometimes referred to as? 8. How can engineering and design methods be characterized? 9. Why is attention to human-machine interaction important?

* Three Mile Island – «Трі Майл Айленд» (атомна електростанція на однойменному острові в штаті Пенсільванія, де у березні 1979 року сталася серйозна аварія внаслідок помилок персоналу та несправності обладнання)

Exercise 21. *Make up questions to the italicized parts of the sentences.*

1. Human–computer interaction is regarded as the intersection of ***computer science, behavioural sciences and design***. 2. ***Interaction between users and computers*** occurs ***at the user interface***, which includes ***both software and hardware***. 3. An important facet of HCI is ***the securing of computer user satisfaction***. 4. ***Engineering and design*** methods are also relevant. 5. Due to ***the multidisciplinary nature of HCI, people with different backgrounds*** contribute to its success.

Exercise 22. *Give nouns corresponding to the following verbs. Translate them.*

Use, receive, define, compute, surround, evaluate, implement, secure, satisfy, develop, communicate, contribute, refer, interact, lead, expect, conclude.

Exercise 23. *Write derivatives of the verbs below and explain their meanings.*

Use, receive, define, compute, surround, evaluate, implement, secure, satisfy, develop, communicate, contribute, refer, interact, lead, expect.

Exercise 24. *Give definitions of the following terms.*

Human–computer interaction, hardware peripherals, large-scale computerized system, human-machine interface, computer graphics, operating system, programming languages.

Exercise 25. *Decipher the abbreviations below.*

HMI, HCI, MMI, CHI, PC, FPD, LCD, RAM, I/O, SCSI, USB, LAN, PoE, EMI, RFI, NEMA.

Exercise 26. *Find some additional information and speak on:*

1. Types and functions of human-machine interface.
2. Human–computer interaction.
3. User interfaces in computing.

Exercise 27. *Use the proper form of the Perfect or Perfect-Continuous Tense of the verbs in brackets.*

1. They (gather) data since last month. 2. He said that he (search) for any valuable information all day yesterday. 3. When they finish

I (wait) for them for thirty minutes. 4. I knew that he (operate) this department for three years. 5. I (analyze) the effectiveness of advertising for two hours when the boss came. 6. Individual end-users (develop) these databases day after day for years. 7. He (enforce) databases security for ten years, why don't you ask him for help? 8. Next week they (protect) their databases from unauthorized activity for a month. 9. He (manage) his project since last month? 10. I wanted to know whether they (try) a new method when you called them? 11. You (work) here for fifteen years by the time you retire? 12. I thought he (work) with unauthorized activity for a long time before. 13. By next semester I (study) this science for four years. 14. For two hours now we (choose) right techniques and we (not come) to any agreement yet. 15. I wondered if they (use) LCD's for a long time. 16. He (repair) peripheral devices for five years on Monday. 17. A new specialist (try) to add a variety of features to his device since early morning. 18. He mentioned that he (implement) interactive computing systems for human use. 19. I (teach) engineering and design methods for two hours when you only enter the university. 20. I asked him if he (work) as a programmer since he graduated from university. 21. He still (improve) his skills for a week? 22. What you (do) with this device? It isn't working! 23. They (try) to eliminate the need for separate power supplies before I came. 24. Since he started his work, he (contribute) to the general success due to his exceptional skills. 25. How long they (investigate) the accident before they left. 26. He (write) his report for three hours when she arrives. 27. I (not repair) my media player yet. 28. They (take) computer based training course before they started teaching. 29. The users (interact) for hours when he arrived. 30. I wonder what he (do) with this keypad so long.

Exercise 28. *Put the verbs in brackets into the proper Perfect or Perfect Continuous form.*

1.My friend is in his laboratory. He (work) hard on his experiment since early morning. 2. I knew that since that time they (apply) these paradigms in such areas as engineering and telecommunications. 3. The employees (prepare) their reports for two days when the head of the department arrives. 4. I wonder what you (discuss) since two o'clock. 5. "Don't worry, they (work) with such equipment for a long time," he said. 6. He found out that I already (make) an attempt to avoid the overhead. 7. He (head) this department for two years when I began working

in it. 8. They (develop) the software for a long period of time. 9. “Look attentively! He (try) to solve this task in his own way.” 10. He (look through) his report several times before he realized his mistake. 11. He said that they (produce) this equipment since 2005. 12. In 2012 she (teach) programming at university for ten years. 13. He (install) a software driver since morning? 14. He asked why I (study) the assignment for so many hours.15. I found out that he (work) there for ten years.

Exercise 29. *Use the proper tense and voice form of the verbs in brackets.*

1. I’m proud that malicious individuals (not penetrate) yet our well-designed computer system. 2. She said that she (take advantages) of this approach for a long period of time. 3. A serious bug already (find) in the program. 4. He (observe) this malfunction since morning. 5. The lecturer explained us that the denial of service attacks (not use) to gain unauthorized access or control of a system. 6. I’m sure all types of attacks (prevent) successfully by next month owing to his experience. 7. They told us that they already (analyze) all possible human errors. 8. By next year he (work) in this department for ten years. 9. I must admit that I (not find) any deception in his activity. 10. What kind of problem (find) lately? 11. I (find) some vulnerabilities very quickly. 12. By tomorrow they (work) on this project for a month. 13. They (download) large quantities of data onto backup media since yesterday. 14. I never (encrypt) the storage media. 15. Some unsuccessful attempts (make) before he came. 16. This kind of software (design) before he knew about it. 17. You (achieve) your goals by next semester? 18. In three years’ time he (accomplish) this work. 19. Some malicious programs just (detect) by the administrator. 20. By nine o’clock he already (install) the program. 21. We (try) to make the computer work since five o’clock. 22. He said that the infection (transmit) by e-mail or Microsoft word documents. 23. I think that you (not get) the consent before I come. 24. The malicious software (infect) the worm into user’s local networks for years. 25. It was clear that he (try) to avoid detection and disinfection for a long time.

Exercise 30. *Change the sentences into indirect speech.*

1. I asked my teacher, “How long have you been teaching this subject?”
2. The manager said, “I have just looked through your personal data.”
3. My friend told me, “I have been trying to enforce the security of my

system through access control." 4. They asked me, "Where did you get the experience in this work?" 5. She told them, "I have been studying object-oriented programming for two years." 6. He asked, "Did you give a right definition to this term last time?" 7. We wondered, "Who managed to solve these difficult problems?" 8. The teacher asked me, "Did you complete your assignment?" 9. The dean said, "You have already proved your serious attitude to studies." 10. My colleague exclaimed, "This is the best job I have ever dreamt of!"

Exercise 31. *Translate the sentences into English. Mind the sequence of tenses.*

1. Ми знаємо, що він займається модернізуванням програмного забезпечення вже впродовж багатьох років. 2. Я думав, що шкідливі програми виявлені. 3. Вона каже, що обов'язково використає резервні носії до кінця тижня. 4. Ми не знаємо, чи вони вже отримали цю порцію даних. 5. Цікаво, чому ці файли виявляють так довго. 6. Я помітив, що він раніше вже запускав такі програми. 7. Я впевнена, що мій колега виявить це порушення до вечора. 8. Ми бачили, що вона знайшла потрібний метод вирішення складного завдання. 9. Викладач пояснив, що він раніше вже успішно використовував такі методики практично. 10. Він повідомив, що наступного року виповниться 20 років, як він працює системним аналітиком. 11. Я була певна, що вони використовують розширювану мову гіпертексту вже довгий час. 12. Вони запитали, коли я отримав доступ до цієї бази даних. 13. Нас вчора повідомили, що такі випробування раніше проводились досить часто. 14. Він обіцяє, що вивчить цю обчислювальну систему до кінця тижня. 15. Ми дізналися, що наші колеги підтримували реалізацію його сумнівного проекту. 16. Я маю сумніви, що вона вже ознайомилась з конфігурацією бази даних. 17. Я хочу знати, що вони вже спланували на наступний навчальний рік. 18. Він здивувався, що ми так довго вивчали новий матеріал. 19. Вона запитала мене, де я знайшов інформацію про цю систему. 20. Він певен, що ми отримаємо пільги та допомогу до наступного місяця.

Exercise 32. *Study the vocabulary to text 3.*

User interface – інтерфейс користувача (*програми*)
aid – 1) допомога; 2) допомагати, сприяти
operational decision – оперативне рішення

interactive aspects – інтерактивні аспекти
applicable – придатний, застосовний
ergonomics – ергономіка, вивчення трудових процесів і умов праці
manipulate – маніпулювати, керувати, поводитися (*з машиною*)
human-machine interaction engineering – технологія взаємодії людини з машиною
relative – відносний, порівняльний, відповідний
decline – зниження, занепад
societal awareness – соціальна (*суспільна*) поінформованість, «соцієтальна» (*широка суспільна*), обізнаність
take on – набувати нового значення
industrial control panel – панель керування, панель приладів
control design – розрахунок системи автоматичного регулювання
textual – текстовий; що стосується тексту
auditory – слуховий
sequence – послідовність
keystroke – натискання клавіши чи кнопки
touch screen – сенсорний екран

Exercise 33. *Translate the word combinations below into Ukrainian.*

Industrial design field; user interface; interaction between humans and machines; effective operation and control of the machine; feedback from the machine; to make operational decisions; broad concept; heavy machinery operator controls and process controls; design considerations; to interact with a machine; hardware (physical) and software (logical) components; to exist for various systems; to indicate the effects of the users' manipulation; the goal of human-machine interaction engineering; to make it easy, efficient, enjoyable to operate a machine; to produce the desired result; to provide minimal input; to achieve the desired output; to minimize undesired outputs to the human; the increased use of personal computers; the relative decline in societal awareness of heavy machinery; keystrokes with the computer keyboard, movements of the computer mouse, and selections with the touch screen.

Exercise 34. *Read and translate text 3.*

Text 3. USER INTERFACE

In the industrial design field of human-machine interaction, the user interface is a place where interaction between humans and machines

occurs. The goal of interaction between a human and a machine at the user interface is effective operation and control of the machine, and feedback from the machine which aids the operator in making operational decisions. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls and process controls. The design considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology.

A user interface is the system by which people (users) interact with a machine. The user interface includes hardware (physical) and software (logical) components. User interfaces exist for various systems, and provide a means of:

- Input, allowing the users to manipulate a system, and/or
- Output, allowing the system to indicate the effects of the users' manipulation.

Generally, the goal of human-machine interaction engineering is to produce a user interface which makes it easy, efficient, enjoyable to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output, and also that the machine minimizes undesired outputs to the human. Ever since the increased use of personal computers and the relative decline in societal awareness of heavy machinery, the term user interface has taken on overtones of the (graphical) user interface, while industrial control panel and machinery control design discussions more commonly refer to human-machine interfaces.

In computer science and human-computer interaction, the user interface (of a computer program) refers to the graphical, textual and auditory information the program presents to the user, and the control sequences (such as keystrokes with the computer keyboard, movements of the computer mouse, and selections with the touch screen) the user employs to control the program.

Exercise 35. *Find in text 3 the English for:*

взаємодія людини з машиною; інтерактивні аспекти комп'ютерних операційних систем; включати такі дисципліни, як ергономіка і психологія; керувати системою; (графічний) інтерфейс користувача; панель приладів; комп'ютерні науки; графічна, текстова і слухова інформація; послідовність управління.

Exercise 36. *Answer the questions on text 3.*

1. What is called the user interface? 2. What is the goal of interaction between a human and a machine at the user interface? 3. What do examples of broad concept of user interfaces include? 4. What are the design considerations applicable to creating user interfaces related to? 5. What physical and logical components does the user interface include? 6. What does the user interface provide? 7. What is a main task of human-machine interaction engineering? 8.What is the difference between (graphical) user interface and human-machine interfaces? 9. What does the user interface of a computer program refer to in computer science and human-computer interaction?

Exercise 37. *Fill in the blanks with prepositions* ***to, in, at, of, with, by, between, from*** *where necessary.*

1. A user interface is the system ... which people interact ... a machine. 2. The goal ... interaction ... a human and a machine ... the user interface is effective operation and control … the machine, and feedback ... the machine which aids the operator ... making operational decisions. 3. Ever since the increased use … personal computers and the relative decline ... societal awareness ... heavy machinery, the term user interface has taken on overtones ... the (graphical) user interface, 4. ... computer science and human-computer interaction, the user interface ... a computer program refers ... the graphical, textual and auditory information the program presents ... the user. 5. Control sequences refer … keystrokes ... the computer keyboard, movements ... the computer mouse, and selections ... the touch screen the user employs to control the program.

Exercise 38. *Write derivatives of the verbs below and explain their meanings.*

Interact, design, effect, operation, decide, consider, apply, vary, manipulate, enjoy, general, desire, aware, society, machine, discuss, present, move, common, industry, select, engineer, graph.

Exercise 39. *Give definitions of the terms below.*

Human-machine interaction, user interface, computer operating system, hardware and software components, human-machine interaction engineering, personal computer, graphical user interface, touch screen.

Exercise 40. *Compose a dialogue on "User Interface".*

Exercise 41. *Speak on user interfaces in computing and the goal of human-machine interaction engineering.*

UNIT 9. MULTIMEDIA

Exercise 1. *Study the basic vocabulary.*

a) terms

medium (*pl.* media) – 1) засіб 2) середовище 3) носій (*інформації*)

media – 1) будь-яка форма інформації *(аудіо-, відеоінформація, анімація тощо)* 2) аудіовізуальне середовище

multimedia – мультимедійні засоби, мультимедіа

hypertext – гіпертекст *(1) технологія, що забезпечує пошук заданих тем у текстових масивах; пошук забезпечується включенням у тексти спеціальних покажчиків, що звуться гіпертекстовими посиланнями (hyperlinks) 2) текст, що має такі гіперпосилання)*

hand-produced material – матеріал, що виробляється ручним способом

still image – статичне зображення

processing device – пристрій оброблення інформації

live performance – вистава наживо

live presentation – презентація наживо

hypermedia – гіпермедіа, гіперсередовище *(розширений, порівняно з гіпертекстом, метод організації мультимедіа-інформації, у якому, крім тексту, підтримуються перехресні посилання з іншими типами даних (відео, графіка, звук)*

linear – лінійний

non-linear – нелінійний

navigational control – навігаційний контроль

viewer – глядач

self-paced – такий, що дозволяє самостійно обирати швидкість вивчення матеріалу *(про учбовий курс)*

computer-based training – комп'ютеризоване навчання

recorded presentation – записана презентація

interactivity – інтерактивність

view in person – переглядати особисто

media player – медіаплеєр

b) nouns
combination – поєднання
animation – анімація
power – сила, потужність
graphics – графіка
presenter – ведучий
performer – виконавець

c) verbs
experience – випробувати *(на собі)*, відчувати; *тут:* споживати
consider – розглядати
reside – *тут:* полягати
offer – пропонувати
view – розглядати

d) adjectives
particular – конкретний
similar – подібний

e) adverbs
broadly – у загальних рисах
locally – локально, у певних межах або масштабах; *тут:* у незначному колі

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Linear, consider, audience, similar, broadly, number, local, allow, interactivity, medium, via, particular, application, typewriter, add, prior, image, employ, huge, option, navigator.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) combination, contrast, traditional, display, device, store, consider, power, reside, subject, divide, live, allow, interaction, view, via, experience, offer, particular, broadly, media, application, locally;

b) different, include, live, power, allow, locally, broadly, divide, still, particular.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: inform – information – informativity – informer –informative

Inform, combine, differ, tradition, record, process, perform, broad, present, train, navigate, local, apply, view, compute, play, produce, animate.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

Media and content; to use traditional forms of printed material; to be recorded and played; computerized and electronic devices; to describe electronic media; to be considered; to reside in hypertext; to jump to another screen; to work with sound, graphics and video; without any navigational control; an example of non-linear content; to be live and recorded; to allow interactivity; via navigation system; a combination of different content forms; in contrast to; hand-produced material; a combination of text, audio, still images, animation, video; to be accessed by information content processing devices; to be a part of a live performance; to store and experience multimedia content; to click on a hypertext word; to be broadly divided into.

Exercise 6. *Read and translate text 1.*

Text 1. Multimedia Content Forms

Multimedia is media and content that uses a combination of different content forms. The term is used in contrast to media which only use traditional forms of printed or hand-produced material. Multimedia includes a combination of text, audio, still images, animation, video, and interactivity content forms.

Multimedia is usually recorded and played, displayed or accessed by information content processing devices such as computerized and electronic devices, but can also be part of a live performance.

Multimedia also describes electronic media used to store and experience multimedia content. Hypermedia can be considered as one particular multimedia application.

The power of multimedia resides in hypertext if you click on a hypertext word, you jump to another screen with more information about that subject. Hypermedia is similar, but also works with sound,

graphics and video. Multimedia may be broadly divided into linear and non-linear categories. Linear active content progresses without any navigational control for the viewer such as a cinema presentation. Non-linear content offers user interactivity to control progress as used with a computer game or used in self-paced computer-based training. Hypermedia is an example of non-linear content.

Multimedia presentations can be live or recorded. A recorded presentation may allow interactivity via a navigation system. A live multimedia presentation may allow interactivity via an interaction with the presenter or performer. They may be viewed in person, or played locally with a media player.

Exercise 7. *Find in text 1 the English for:*

поєднання різних форм контенту; на відміну від; використовувати традиційні форми; друкований матеріал; матеріал, що виробляється ручним способом; поєднання тексту, аудіо та статистичних зображень; записуватися та програватися; може розглядатися як конкретне мультимедійне застосування; комп'ютеризовані та електронні пристрої; презентація наживо; споживати мультимедійний контент; більше інформації із цього питання; потужність мультимедійних засобів полягає у гіпертексті; в загальних рисах; без будь-якого навігаційного контролю; забезпечувати користувачеві інтерактивність; комп'ютеризоване навчання; презентації можуть вестися наживо або зберігатися у записі; з допомогою навігаційної системи; через взаємодію з ведучим або виконавцем; переглядати особисто; програватися з допомогою медіаплеєра.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. Multimedia is media that uses a combination of similar content forms. 2. Multimedia uses traditional forms of printed or hand-produced material. 3. Multimedia is only recorded by information content processing devices. 4. Multimedia also describes electronic media. 5. Hypermedia can't be considered one particular multimedia application. 6. The power of multimedia resides in hypertext. 7. Hypermedia works with graphics. 8. Multimedia may be divided into three categories. 9. Linear active content progresses with navigational control.

10. Hypermedia is an example of linear content. 11. Multimedia presentations can be live or recorded. 12. A recorded presentation may allow interactivity without navigation system. 13. A live multimedia presentation may allow interactivity via an interaction with the viewer. 14. Live multimedia presentations may be viewed in person.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1. to use	a) electronic media
2. to describe	b) sound, graphics and video
3. to offer	c) multimedia content
4. to work with	d) another screen
5. to allow	e) linear and non-linear categories
6. to experience	f) information content processing devices
7. to jump to	g) interactivity via a navigation system
8. to be divided into	h) printed or hand-produced material
9. to include	i) interactivity to control progress
10. to be accessed by	j) a combination of text, audio, images

Exercise 10. *Fill in the blanks with prepositions* ***for, on, to, of, about, by, in, with, without, via, into*** *where necessary.*

1. The term is used … contrast … media which only use traditional forms ... material. 2. Multimedia is usually recorded and played, displayed or accessed ... information content processing devices. 3. The power … multimedia resides ... hypertext. 4. If you click … a hypertext word, you jump … another screen … more information … that subject. 5. Multimedia may be broadly divided ... linear and non-linear categories 6. Linear active content progresses … any navigational control … the viewer. 7. Non-linear content offers user interactivity to control progress as used ... a computer game. 8. A recorded presentation may allow interactivity ... a navigation system. 9. They may be viewed … person, or played locally … a media player.

Exercise 11. *Fill in the blanks with proper terms (****hypermedia, multimedia (2), hypertext, linear category, non-linear category, recorded presentation, live presentation, media player****) to complete the sentences.*

1. _____ allows to play locally. 2. _____can be considered one particular multimedia application. 3. _____ allows interactivity via navigation system. 4. _____ is the power of multimedia. 5. _____ allows interactivity via an interaction. 6. _____ offers interactivity to control progress. 7. _____ uses a combination of different content forms. 8. _____ works without any navigational control. 9. _____ is recorded and played by information processing devices.

Exercise 12. *Answer the questions on text 1.*

1. What is multimedia? 2. What does multimedia include? 3. What is multimedia usually recorded, played, displayed or accessed by? 4. What can multimedia describe? 5. What is called hypermedia? 6. What does the power of multimedia reside in? 7. What happens if you click on a hypertext word? 8. What does hypermedia work with? 9. What categories may multimedia be divided into? 10. What are linear and non-linear contents? 11. What does non-linear content offer? 12. What can be live or recorded? 13. What may a recorded presentation allow? 14. How can a recorded and live presentation be experienced?

Exercise 13. *Put all possible questions to the sentences below.*

1. Multimedia uses a combination of different content forms. 2. Multimedia also describes electronic media devices used to store and experience multimedia content. 3. Multimedia may be broadly divided into linear and non-linear categories. 4. Non-linear content offers user interactivity to control progress. 5. A live multimedia presentation may allow interactivity via an interaction with the presenter or performer.

Exercise 14. *Translate into English.*

1. Мультимедіа – це будь-яка форма інформації, що використовує поєднання різних форм контенту. 2. Цей термін відрізняється від поняття середовища, що використовує лише звичайні форми друкованого матеріалу або матеріалу, що виробляють ручним способом. 3. Мультимедіа поєднує різні форми контенту, такі як текст, аудіо, статичні зображення, анімацію, відео та інтерактивність. 4. Мультимедійні засоби записуються, програються, зображаються та викликаються пристроями оброблення інформаційного контенту. 5. Мультимедійні засоби є також електронними засобами, які використовують для зберігання та споживання мультимедійного контенту.

6. Гіпермедіа може вважатися конкретним застосуванням мультимедійних засобів. 7. Потужність мультимедійних засобів полягає в гіпертексті. 8. Гіпермедіа також працює зі звуком, графікою та відео. 9. Мультимедійні засоби можна поділити на лінійну та нелінійну категорії. 10. Лінійний активний контент працює без будь-якого навігаційного контролю. 11. Нелінійний контент забезпечує інтерактивність керування процесом, що використовується в комп'ютерних іграх. 12. Гіпермедіа – це приклад нелінійного контенту. 13. Мультимедійні презентації можуть відбуватися наживо або у записі. 14. Записані презентації уможливлюють інтерактивність з допомогою навігаційної системи. 15. Під час прямих мультимедійних презентацій інтерактивність забезпечується через взаємозв'язок з ведучим або виконавцем. 16. Вони можуть розглядатися особисто або програватися з допомогою медіаплеєра.

Exercise *15. Write a summary of the text "Multimedia Content Forms".*

Exercise 16. *Study the vocabulary to text 2.*

Denote – 1) означати; 2) позначати
property – властивість
framework – структура
collaborate – співпрацювати
affordable – доступний
previously – раніше
exponential decline – експонентне зниження / зменшення
enable – дозволяти
overtake – 1) наздогнати, надолужити; 2) випередити
breeding ground – родючий ґрунт
delivery vehicle – засіб доставлення
multimedia application – 1) мультимедійна програма; 2) застосування мультимедійних засобів
appealing – привабливий
storehouse of information – енциклопедія
media control tools – засоби керування аудіовізуальним середовищем
pleasing – приємний
suited – придатний, підхожий
prior – попередній
recent – останній, новітній

Exercise 17. *Read, translate and entitle text 2.*

Text 2

The term multimedia is used to denote the property of handling a variety of representation media in an integrated manner. It is also important that the various sources of media types are integrated into a single system framework. Multimedia adds interactivity to the combination of text, graphics, images, audio and video. Creating your own media is more interactive than using existing content, and collaborating with others in the creation of media is still more interactive. Multimedia has been used for education at all levels, job training, and games and by the entertainment industry.

Multimedia as a human-computer interface was made possible some half-dozen years ago by the rise of affordable digital technology. Previously, multimedia effects were produced by computer-controlled analogue devices. Digital technology's exponential decline in price and increase in capacity has enabled it to overtake analogue technology.

The Internet is the breeding ground for multimedia ideas and the delivery vehicle of multimedia objects to the huge audience. Any computer application that employs video disk, images from a CD-ROM, uses high quality sound, or uses high quality video images on screen may be termed a multimedia application. Such interfaces are often aesthetically appealing and, where high capacity storage devices such as CD-ROM are used, can provide effective interaction for the user by acting as very large databases or storehouses of information. A multimedia user interface must provide a wide variety of easily understood and usable media control tools. The interface design should be aesthetically pleasing, appropriate to the content and suited to the learner's culture and prior knowledge. The development of graphical user interfaces and recent advances in the field of virtual reality allow users to control the system by manipulating objects as icons, windows, menus etc.

Exercise 18. *Put some key questions on text 2.*

Exercise 19. *Find in text 2 the English for:*

використовуватися для визначення; властивість інтегровано працювати з різноманітними засобами представлення; бути інтегрова-

ним в одній системній структурі; додавати інтерактивності; створення власного середовища; використання наявного контенту; співпрацювати з іншими; індустрія розваг; доступна цифрова технологія; аналогові пристрої, що керуються комп'ютером; експонентне зниження; перевершити аналогову технологію; родючий ґрунт; засіб доставлення; велика аудиторія; високоякісний звук; можна назвати мультимедійною програмою; забезпечувати ефективну взаємодію; естетично привабливий; відповідати контенту; попередні знання; останні досягнення в галузі віртуальної реальності; керувати системою шляхом маніпулювання об'єктами.

Exercise 20. *Memorize the forms of the Infinitive.*

	Active	Passive
Indefinite (Simple)	to do	to have done
Continuous	to be doing	------
Perfect	to have done	to have been done
Perfect Continuous	to have been doing	------

Exercise 21. *Translate the sentences. Define the forms and functions of the Infinitive. (It may be used as a subject, an attribute, an object, a part of a predicate, an adverbial modifier).*

1. Multimedia is the use of several different media to convey information. 2. Multimedia can be referred to computer media. 3. A media stream can be on demand or live. 4. Attempts to display media on computers date back to the earliest days of computing. 5. On demand streams can be stored on a server for a long period of time, and are available to be transmitted at a user's request. 6. Individual printers can be designed to support both local and network connected users. 7. The code must be written in time. 8. The students believe to be helped by the teacher. 9. I was pleased to have met her at the conference. 10. We are glad to be working on this project. 11. He is glad to have known the professor for many years.12. Software can be stored on a hard drive or flash drive. 13. To run a program, the computer is to read the instructions and temporarily place them in random-access memory (RAM) before they can be executed.

Exercise 22. *Complete the sentences using the proper form of the Infinitive in brackets.*

1. In education multimedia is used (to produce) computer-based training courses. 2. In mathematical and scientific research multimedia can (to use) for modelling and simulation. 4. The students are glad (to pass) the exam. 5. Multimedia may (to divide) into linear and non-linear categories. 6. The participants were glad (to tell) a lot about the research. 7. He seems (to program) since morning. 8. I am sorry (to miss) the lecture. 9. I am glad (to do) all the homework yesterday. 10. He seems (to solve) this problem now. 11. I want (to take) to the conference by my friend. 12. They are supposed (to work) on this project for the last two years. 13. He seems (to learn) all programming languages. 14. Programming can (to perform) using a step-by-step process composed of coding, debugging and testing. 15. He hopes (to improve) his English at the University. 16. The instructions tell the computer what operations (to perform) with the data. 17. The instructions must (to translate) into a machine language.

Exercise 23. *Translate the sentences paying attention to the Objective Infinitive Complex.*

1. Everybody knows Bill Gates to be one of the richest and most successful people in the world. 2. People know Microsoft to be the world's largest software company. 3. I saw him reinstall the system. 4. I noticed her enter the study. 5. We expected him to solve the problem. 6. We watched the teacher approach the University. 7. My parents wanted me to become a software engineer. 8. The teacher would like the students to speak English.

Exercise 24. *Translate the sentences paying attention to the Subjective Infinitive Complex.*

1. Altair is known to be one of the world's first microcomputers. 2. Harvard is known to be America's most famous university. 3. Bill Gates happened to be one of the world's youngest billionaires at the age of 31. 4. At school he proved to be very intelligent, especially good at Math. 5. Bill Gates is known to have always been very successful and hardworking. 6. He seems to know a lot about engineering software. 7. We happened to meet them at the conference. 8. The developer seems

to have been working since morning. 9. Little progress seems to have been made for several decades, primarily due to the high cost and limited capabilities of computer hardware.

Exercise 25. *Paraphrase the sentences using the Objective Infinitive Complex.*

Model:	They think that these exercises are very difficult. – They think **these exercises to be very difficult**.

1. I heard that the door of the entrance hall opened. 2. My wish is that you should enter this university. 3. His father wants that he will become a programmer. 4. I know that he is a very experienced teacher. 5. I know that he works at this software company. 6. Do you expect that a contract will be signed tomorrow? 7. I don't consider that they are right. 8. I know that our teacher is an excellent mathematician. 9. He heard that the students spoke in low voices. 10. They supposed that the printer will operate properly. 11. They found that the code was too complicated.

Exercise 26. *Paraphrase the sentences using the Subjective Infinitive Complex.*

Model:	It was announced that our group would be the next. – **Our group** was announced **to be the next**.

1. It turned out that she completed the laboratory work yesterday. 2. They will win the competition. We are sure of it. 3. People consider the problem to be very difficult. 4. It seems that she knows a lot about developing software. 5. It is said that this topic is very important. 6. They consider that this device is necessary for their experiments. 7. I expect that he will solve the problem very quickly. 8. We expect that you will debug this program. 9. It is known that these exercises are very useful. 10. It seems that she speaks English perfectly. 11. They believe that the teacher will help them. 12. They expect that he will take the first prize.

Exercise 27. *Study the vocabulary to text 3.*

Enhance – поліпшувати, збільшувати
distract – заважати, відволікати
interpret – тлумачити, пояснювати, інтерпретувати

read back – 1) зчитувати щойно записану інформацію; 2) повторно зчитувати
reasonable – прийнятний, підхожий, адекватний
fidelity – точність
vision-impaired – з вадами зору
compared to – порівняно (*з чимось*)
lack – бракувати, не вистачати
digitized – (п)оцифрований
synthesized – синтезований
relative timing – відносна синхронність, узгодженість у часі
caption – субтитр
convey – передавати *(думку)*, висловлювати (*думку*)
sense – відчуття
frame – кадр
temporal relationships – часові співвідношення
successive – послідовний
capture – фіксування зображення

Exercise 28. *Translate the word combinations below into Ukrainian.*

To play an important part; highly distracting; storage of images; text-to-speech technology; in the same way as; with reasonable fidelity; large amount of text; without large sound files; vision-impaired people; compared to human speech; make human speakers appealing; to provide high fidelity; successive capture; improved techniques for generating speech; the message is converted by the motion and the sense.

Exercise 29. *Read and translate text 3.*

Text 3. Sound and Images in Multimedia

Just as video has a role in multimedia, sound also plays an important part. A few carefully placed sounds can greatly enhance a project, but a continuous monologue can be highly distracting. With the text-to-speech technology, the computer interprets text and converts it into phonetic sounds in much the same way as a human would. Thus, the computer can read back any text within any program with reasonable fidelity. This feature is very useful because large amounts of text can be converted into audio without large sound files. A particular use of this technology is to offer an alternative for vision-impaired people. There

are however, some disadvantages to computer generated speech. The speech can sound robotic compared to human speech and it lacks the variable information that can make human speakers appealing. Two types of speech are available for use by multimedia developers: digitized and synthesized. Digitized speech provides high quality natural speech while synthesized speech may not sound as natural as human speech. Even with improved techniques for generating speech, it is not incorporated into multimedia programs as often as it could be.

Unlike print or graphics, animation is a dynamic medium. We get a sense of relative timing, position, direction and speed of action. We need no captions because the message is converted by the motion and the sense. Animation is the process of creating, usually graphically, a series of frames and then having them display rapidly to get a sense of movement. Video provides high-speed information transfer and shows temporal relationships. Video is produced by successive capture and storage of images as they can change with time.

Exercise 30. *Find in text 3 the English for:*

так само як і відео; відігравати важливу роль; покращити якість проекту; дуже заважати; технологія перетворення тексту на мовлення; комп'ютер інтерпретує та перетворює; ця властивість є дуже корисною; конкретне застосування цієї технології; пропонувати альтернативу; робити мову привабливою; такий природний, як людська мова; на відміну від друку або графіки; динамічне середовище; відчувати відносну узгодженість у часі; напрям і швидкість дії; повідомлення передається рухом і відчуттям; процес створення послідовності кадрів; часові співвідношення; високошвидкісне передання інформації; фіксування та зберігання відеозображень.

Exercise 31. *Answer the questions on text 3.*

1. What role does sound play in multimedia? What can it do? 2. How does the computer interpret and convert text? 3. What feature is considered to be very useful and what does it offer? 4. What disadvantages does computer generated speech have? 5. What two types of speech are available for use by multimedia developers? 6. What is the difference between them? 7. What is animation? 8. What is an animated message conveyed by? 9. What can video provide? 10. What is video produced by?

Exercise 32. *Write derivatives of the words below and explain their meanings.*

Enhance, interpret, convert, use, generate, compare, digitize, synthesize, improve, incorporate, create, produce.

Exercise 33. *Find some additional information and speak on:*

1. Hypermedia.
2. Multimedia as a human-computer interface.
3. Sound and graphics in multimedia.

Exercise 34. *Study the vocabulary to text 4.*

Bitmap – растр, растрове (*бітове*) зображення *(графічне зображення у вигляді масиву точок на екрані)*
bitmapped graphics – растрова графіка
distorted – викривлений
resolution – роздільна здатність
inch – дюйм
vector graphics – векторна графіка
scale – масштабувати
facilities – можливості
freehand drawing – малювання рукою
pie chart – секторна, кругова діаграма
bar chart – стовпчикова діаграма, гістограма
computer-aided design – автоматизоване проектування
desktop publishing – верстка друкованих видань на комп'ютері
page layout program – програма компонування сторінок
clip-art – політипаж, графічний фрагмент
applet – прикладна міні-програма, утиліта
fractal – фрактал *(геометрична форма, яка може бути розбита на окремі частини, котрі є приблизними зменшеними копіями цілого; фрактали описують такі об'єкти реального світу, як: гори, контури берегів, хмари тощо)*

Exercise 35. *Translate the word combinations below into Ukrainian.*

Pictures processed by computers; a bit distorted; the density of dots; without losing quality; to edit your favourite images; to add different effects; facilities for freehand drawing; computer-aided design; desktop

publishing; to arrange on a page; to create or edit moving pictures; detailed maps.

Exercise 36. *Read and translate text 4.*

Text 4. Computer Graphics

Computer graphics are pictures created, changed or processed by computers. There are two categories**:** bitmapped graphics and vector graphics. Bitmapped graphics represents images as bitmaps; they are stored as pixels and can become a bit distorted when they are manipulated. The density of dots, known as the resolution and expressed in dots per inch, determines how sharp the image is. Vector graphics represents images as mathematical formulae, so they can be changed or scaled without losing quality.

There are different types of graphics software. Image manipulated programs let you edit your favourite images. For example, you can scan a picture into your PC or transfer a photo from your camera and add different effects, or filters. Painting and drawing programs, also called illustration packages, offer facilities for freehand drawing. Business graphics programs, also called presentation software, let you create pie charts and line graphs of all kinds for slide shows and reports. Computer-aided design (CAD) is used by engineers and architects to design everything from cars and plants to buildings and furniture. Desktop publishing (DTP) is based around a page layout program, which lets you import text from a word processor, clip-art (ready-made picture) from graphics packages, and images from scanners or cameras, and arrange them all on a page. Digital art, or computer art, is done with applets that use mathematical formulae to create beautiful bright shapes called fractals. A fractal is a geometrical figure with special properties. Computer animation uses graphics programs to create or edit moving pictures. Each image in a sequence of images is called a ‘frame’. Geographic information systems (GIS) allow cartographers to create detailed maps.

Exercise 37. *Find in text 4 the English for:*

растрова графіка; зберігатися у вигляді пікселів; щільність точок; визначати, наскільки чітким є зображення; математичні зображення; сканувати зображення; створювати секторні та стовпчикові

діаграми; програма компонування сторінок; геометрична фігура зі специфічними властивостями; послідовність зображень.

Exercise 38. *Answer the questions on text 4.*

1. What is computer graphics? 2. How many categories of computer graphics are there? What are they? 3. What kind of graphics represents images as bitmaps? 4. What is resolution in computer graphics? 5. What program edits images? 6. What do painting and drawing programs offer? 7. What are business graphics programs called? 8. What do they create? 9. Who is computer-aided design used by? 10. What is desktop publishing? 11. What is a fractal? Where is it used? 12. What does computer animation use? 13. What do geographic information systems allow to create?

Exercise 39. *Find in text 4 the words that can function both as nouns and verbs. There should be 20 of them.*

Exercise 40. *Decipher the abbreviations below:*

CAD, DTP, GIS, MMI, HCI, CHI, LCD, I/O, Mbps, NEMA.
Exercise 41. *Compose a dialogue on "Computer Graphics".*

UNIT 10. INFORMATION SYSTEMS

Exercise 1. *Study the basic vocabulary.*

a) terms
information system (IS) – інформаційна система
office information system – офісна інформаційна система
transaction processing system – система оброблення транзакцій
management information system – управлінська / адміністративна інформаційна система
decision support system (DSS) – система підтримки рішень
expert system – експертна система
communications technologies – комунікаційні технології
voice mail – голосова пошта; автовідповідач компанії
videoconferencing – відеоконференц-зв'язок
electronic data interchange – електронний обмін інформацією
management reporting system – система управлінської звітності
inference rules – правила виведення *(висновків в експертних системах)*

b) nouns
collection – сукупність
workflow – трудовий процес
business activity – бізнес-операція
office activity – офісна операція, вид офісної діяльності
speaker – гучномовець
deposit – 1)депозит, рахунок у банку; 2) депозитний внесок
payment – сплата, платіж
reservation – резервування, бронювання
inventory – інвентаризація, облік товару
interest rate – ставка банківського відсотка
population trends – тенденції зміни структури та кількості населення
reasoning – мислення, міркування
expertise – знання, кваліфікація, досвід
judgement – 1) судження, думка, погляд, оцінка; 2) здоровий глузд, розсудливість

c) verbs
enhance – поліпшувати, удосконалювати
facilitate – полегшувати
distribute – розподіляти
schedule – складати розклад, планувати
account – здійснювати облік
arise – виникати, з'являтися, поставати
capture – збирати *(інформацію)*
imitate – імітувати

d) adjectives
day-to-day – повсякденний
short-range – короткочасний
long-range – тривалий

e) adverbs
accurately – точно, правильно
timely – своєчасно

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Facilitate, payment, distribute, apply, describe, reservation, internal, timely, accurately, order, collection, imitate, capture, expertise, frequently, deposit, manipulate, devote, unit, resource, retrieve, employee, transmit.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) collection, procedure, generate, day-to-day, activity, classify, variety, equip, reach, decision, capture, store, imitate, expertise, compose, component, apply, describe, support, enhance, facilitate, distribute, arise, workflow, reasoning, regular;

b) employee, variety, source, internal, enhance, facilitate, timely, support, capture, input, accurate, order, frequent, connect, regular, relevant, advantage.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: distribute – distribution – distributor – distributive

Distribute, collect, organize, classify, process, manage, commune, create, accurate, manufacture, judge, decide, report, generate, support, inform, imitate, apply, describe, speak, reserve, time, expert, employ, store, transmit, proceed, busy, act, source, author, change.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

A collection of hardware, software, data, people and procedures; to be designed to generate information; to support day-to-day activity; to be classified into; to facilitate communication among employees; to enhance workflow; electronic data interchange; a variety of hardware; to be equipped with modems, video cameras, speakers; to capture and process data; day-to-day transactions; business activity; online transaction processing; a management information system; to generate reports on a regular basis; to be called a management reporting system; to reach a decision; internal and external sources; to include interest rates; to have less expertise; a knowledge base; inference rules; to describe a situation.

Exercise 6. *Read and translate text 1.*

Text 1. Types of Information Systems

An information system is a collection of hardware, software, data, people and procedures that are designed to generate information that supports day-to-day, short-range, and long-range activities of users in an organization. Information systems are classified into five categories: office information systems, transaction processing systems, management information systems, decision support systems, and expert systems.

An office information system is an information system that uses hardware, software and networks to enhance workflow and facilitate communications among employees. An office information system supports a range of business office activities such as creating and distributing graphics or documents, sending messages, scheduling, and accounting. Office information systems use communications technologies such as voice mail, facsimile (fax), videoconferencing, and electronic data interchange. It also uses a variety of hardware, including computers equipped with modems, video cameras, speakers, and microphones, scanners, and fax machines.

A transaction processing system is an information system that captures and processes data generated during an organization's day-to-day transactions. A transaction is a business activity such as a deposit, payment, order or reservation. Most transaction processing systems use online transaction processing.

A management information system is an information system that generates information accurately and timely. Because it generates reports on a regular basis, a management information system is called a management reporting system.

A decision support system is designed to help users reach a decision when a decision-making situation arises. It uses data from internal or external sources. Internal sources include sales, manufacturing, inventory. External sources include interest rates, population trends.

An expert system is an information system that captures and stores the knowledge of human experts and then imitates human reasoning and decision-making processes for those who have less expertise. Expert systems are composed of two main components: a knowledge base and inference rules. A knowledge base is a combined subject knowledge and

experiences of the human experts. The inference rules are logical judgments applied to the knowledge base each time a user describes a situation to the expert system.

Exercise 7. *Find in text 1 the English for:*

інформаційна система; офісна інформаційна система; низка офісних операцій; система оброблення транзакцій; полегшувати зв'язок між працівниками; створення та розподіл графічних даних; відправлення повідомлень; використовувати комунікаційні технології; різноманітні апаратні засоби; бізнес-операція; депозитний внесок; платіж; замовлення; правильно і своєчасно надавати інформацію; тенденції зміни структури та кількості населення; виробництво; інвентаризація; збирати та зберігати інформацію; імітувати мислення людини; складатися з двох компонентів; знання і досвід експертів у певній предметній галузі; логічні судження; експертна система.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. An information system is a collection of software, people and procedures. 2. Information systems are designed to generate information that supports short-range activities of users. 3. Information systems are classified into five categories. 4. An office information system facilitates communication among employees. 5. An office information system doesn't use communications technologies. 6. It uses a variety of hardware. 7. Not all transaction processing systems use online transaction processing. 8. A management information system generates reports on a regular basis. 9. A decision support system uses external sources. 10. External sources include sales, manufacturing, inventory. 11. An expert system stores the knowledge of human experts. 12. Expert systems include three components. 13. A knowledge base means experience of human experts. 14. The inference rules are logical judgements applied to the knowledge base.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to generate	a) users reach a decision
2) to store	b) human reasoning

3) to facilitate	c) modems and video cameras
4) to imitate	d) communications among employees
5) to classify into	e) day-to-day activities
6) to be equipped with	f) knowledge of human experts
7) to include	g) workflow
8) to support	h) categories
9) to help	i) information
10) to enhance	j) sales, manufacturing, inventory

Exercise 10. *Fill in the blanks with prepositions* ***for, as, with, of, to, into, by, on, from*** *where necessary:*

1. Information systems are classified … five categories. 2. An office information system supports a range … business office activities. 3. It also uses a variety … hardware equipped … modems, video cameras, speakers. 4. A transaction is a business activity known … a deposit, payment, order or reservation. 5. Because it generates reports … a regular basis, a management information system is referred ... a management reporting system. 6. A decision support system uses data … internal or external sources. 7. Expert systems are composed … two main components. 8. The inference rules are logical judgements applied … the knowledge base each time a user describes a situation ... the expert system. 9. Business office activities are supported ... an office information system. 10. An expert system imitates human reasoning and decision-making processes ... those who lack expertise.

Exercise 11. *Fill in the blanks with proper terms* ***(office information system (2), internal sources, decision support system, transaction, management information system, information system, expert system, the inference rules****) to complete the sentences.*

1. _____ is a system that generates information accurately and timely. 2. _____ is a business activity such as a deposit, payment, order or reservation. 3. _____ is a system that facilitates communications among employees. 4. _____ is designed to help users reach a decision. 5. _____ is a collection of hardware, software, data, people and procedures. 6. _____ is a system that supports a range of business office activities. 7. _____ is a system

that stores the knowledge of human experts. 8. _____ are logical judgements applied to the knowledge base. 9._____ include sales, manufacturing, inventory.

Exercise 12. *Answer the questions on text 1.*

1. What is an information system? 2. What categories are information systems classified into? 3. What does an information system use and support? 4. What communications technologies do office information systems use? 5. What does a transaction processing system capture and process? 6. What is a transaction? 7. What do most transaction processing systems use? 8. How does a management information system generate information? 9. What does a decision support system help users do? What does it use? 10. What do internal and external sources include? 11. What is called an expert system? 12. What components are expert systems composed of? 13. What is a knowledge base? 14. What are inference rules in an expert system?

Exercise 13. *Put all possible questions to the sentences below.*

1. An office information system supports a range of business office activities. 2. A transaction is a business activity such as a deposit, payment, order and reservation. 3. A decision support system is designed to help users reach a decision. 4. Expert systems are composed of two main components. 5. Office information systems use communications technologies.

Exercise 14. *Translate into English.*

1. Інформаційна система – це сукупність апаратного та програмного забезпечення, інформації, людей та процедур, призначених для оброблення інформації. 2. Інформаційні системи поділяють на п'ять категорій: офісні інформаційні системи, системи оброблення транзакцій, управлінські інформаційні системи, системи підтримки рішень, експертні системи. 3. Офісна інформаційна система використовує апаратне та програмне забезпечення і мережі для вдосконалення робочого процесу. 4. У офісних інформаційних системах використовують такі комунікаційні технології, як: голосова пошта, факс, відеоконференц-зв'язок, електронний обмін інформацією. 6. Транзакція – це бізнес-операція, наприклад: депозитний внесок, платіж, замовлення та резервування.

7. Управлінська інформаційна система призначена для правильного та своєчасного надання інформації. 8. Оскільки управлінська інформаційна система генерує звіти на регулярній основі, її називають також системою управлінської звітності. 9. Система підтримки рішень допомагає користувачам знайти рішення у ситуації його прийняття. 10. Використовуються дані з внутрішніх та зовнішніх джерел. 11. Внутрішні джерела включають продаж, виробництво, інвентаризацію. 12. До зовнішніх джерел відносять ставки банківських процентів, тенденції зміни структури та кількості населення. 13. Експертна система збирає та зберігає знання експертів, імітує мислення людини та процес прийняття рішення для тих, хто має менше досвіду. 14. Експертні системи складаються з двох компонентів: бази знань та правил виведення експертних висновків. 15. База знань – це поєднання знань та досвіду експертів у певній предметній галузі. 16. Правила виведення висновків – це логічні судження й оцінки, що застосовують до бази знань щоразу, як користувач описує ситуацію експертній системі.

Exercise 15. *Write a summary of the text "Types of Information Systems".*

Exercise 16. *Study the vocabulary to text 2.*

In a broad sense – у широкому сенсі
refer – називати, позначати
business process – бізнес-процес, технологічний / виробничий процес
distinction – різниця, відмінність
distinct from – відмінний (*від чогось*)
information and communication technology (ICT) – інформаційно-комунікаційні технології
performance – робота, функціонування
work system – *тут:* виробнича система
customer – клієнт, замовник
devote – присвячувати
retrieve – вибирати *(інформацію з пам'яті)*
manipulate – маніпулювати
socio-technical system – соціальнотехнологічна система
mediating construct – проміжний структурний компонент
interrelate – 1) бути взаємопов'язаним; 2) взаємодіяти
represent – представляти, презентувати

Exercise 17. *Translate the word combinations below into Ukrainian.*

The combination of information technologies and people's activities; to be used to refer to the interaction between people, algorithmic processes, data and technology; the way in which people interact; information systems are distinct from information technology; to help to control the performance of business processes; to perform work using resources; to produce specific products or services for customers; capturing, transmitting, storing, retrieving, manipulating and displaying information.

Exercise 18. *Read, translate and entitle text 2.*

Text 2.

An information system is any combination of information technologies and people's activities using that technology to support operations, management and decision-making. In a very broad sense, the term information system is frequently used to refer to the interaction between people, algorithmic processes, data and technology. In this sense, the term is used to refer not only to the information and communication technology, but also to the way in which people interact with this technology in support of business processes.

There is a clear distinction between information systems and information technology as well as between information systems and business processes. Information systems are distinct from information technology in that an information system is typically seen as having an ICT component. Information systems are also different from business processes. Information systems help to control the performance of business processes.

An information system is a special type of work system. A work system is a system in which humans or machines perform work using resources (including ICT) to produce specific products or services for customers. An information system is a work system whose activities are devoted to processing (capturing, transmitting, storing, retrieving, manipulating and displaying) information.

An information system is a type of socio-technical system and a mediating construct between actions and technology. Information systems interrelate with data systems on the one hand and activity systems on the other. An information system is a form of

communication system in which data are represented and processed as a form of social memory. An information system can also be considered as a semi-formal language which supports human decision making and action.

Exercise 19. *Find in text 2 the English for:*

у широкому сенсі; підтримувати операції керування та прийняття рішень; на підтримку бізнес-процесів; чітка відмінність процесів бізнесу; виробляти товари та послуги; робота, спрямована на оброблення інформації; соціально-технологічна система; проміжний структурний компонент між діями та технологією; бути взаємопов'язаними із системами даних з одного боку, і системами дій, – з іншого; представлятися та оброблятися як одна з форм соціальної пам'яті; розглядати як напівформальну мову.

Exercise 20. *Answer the questions on text 2.*

1. What is an information system? 2. What is the term information system used to refer to in a broad sense? 3. What differs information systems from information technology and business processes? 4. What is called a work system? 5. What activities is an information system devoted to? 6. What place does an information system hold between actions and technology? 7. What do information systems interrelate with? 8. What can an information system be considered as?

Exercise 21. *Memorize the forms of the Participle.*

	Active	Passive
Present	buying	being bought
Perfect	having bought	having been bought
Past	--------	bought

Exercise 22. *Give Present and Past Participle forms of the following verbs:*

Stop, cut, forget, begin, order, travel, study, open, play, develop, write, give, lie, refer, try, profit, compel, carry, repair.

Exercise 23. *Translate the following sentences. Define the forms and functions of the Participle. (It may be used as an attribute, an adverbial modifier, a part of a verbal predicate)*

1. Although accepted for different purposes computers virtually do not differ in structure. 2. While processing data the computer made a few mistakes. 3. Computers today are providing an expending range of services. 4. Having been given all the instructions the designer was able to start his work immediately. 5. Programs stored in the memory of a computer enable the computer to perform a variety of tasks. 6. The model being shown to us now has been made by one of the best specialists of our company. 7. The article on software engineering published in this magazine was written by the Dean of our Faculty. 8. You can get the book recommended by our lecturer in the library. 9. The books lying on the table belong to my roommate. 10. Having signed the letter the project manager gave it to the secretary asking her to send it off at once. 11. What do you think of the method being used? 12. An interpreter is a program translator used for translating high-level language programs into machine language programs. 13. Changing the program, the computer can be made to work in a different manner. 14. Having developed Windows Vista Microsoft Company recommended it for use on personal computers. 15. Application software being presented for an organization is useful for word processing, billing and accounting.

Exercise 24. *Write out all Participles from the text of Exercise 18. Identify their forms and functions.*

Exercise 25. *Point out the Objective Participial, Subjective Participial and Absolute Participial Complexes. Translate the sentences.*

1. Hardware engineer was watched designing ICT devices. 2. I saw him reinstalling the system of his PC. 3. Software engineers were seen planning, designing and testing computer programs. 4. Various strategies and techniques being accessible, security specialists use them to design security systems. 5. She had her computer repaired. 6. The network requirements having been studied, analysts recommended the most suitable type of the network. 7. The successful candidate was considered being responsible for managing the database. 8. She watched

him writing an antivirus program. 9. The data being encrypted, a malicious program cannot corrupt files. 10. He watched the designer creating and maintaining web pages and applications for websites. 11. Computer operator was watched controlling data processing. 12. He had his system assembled. 13. A special gift having been offered, people happily sent off all their security details. 14. I saw the computer instructor teaching people how to protect computer systems and information. 15. Programmers were seen writing the instructions for customized software.

Exercise 26. *Change the following complex sentences into simple ones using the Objective Participial Construction after the model:*

Model:	I saw them as they were going to the University. – I saw **them going** to the University.

1. I saw him as he was reinstalling the system of his PC. 2. We watched them as they were playing computer games. 3. The teacher observed the students as they were writing the test. 4. I saw the programmer as he was setting the computer. 5. She watched him as he was testing a program. 6. We noticed our lecturer as he was entering the hall.

Exercise 27. *Change the following complex sentences into simple ones using the Objective Participial Construction after the model:*

Model:	The system analyst repaired her computer. – She **had** her computer **repaired**.

1. His friend assembled a system for him. 2. The photographer took a photo of him. 3. The typist typed his article for him. 4. The waitress brought them dinner. 5. His students made reports for him. 6. Their lecturer organized an excursion for them. 7. The organizers of the exhibition provided a working place for them. 8. They altered devices for them.

Exercise 28. *Change the following complex sentences into simple ones using Absolute Participial Complex as in the models.*

Model 1:	As the University was far from here, I went there by bus. – **The University being far from here**, I went

Model 2:	there by bus. When the work had been done, they went home. – **The work having been done**, they went home.

1. As his mother teaches English, she knows the language very well. 2. As he was ill, he couldn't go to the University. 3. When our teacher had visited the IT exhibition, we asked him to tell us about it. 4. When the lessons were over, the students went home. 5. After the sun had risen, we continued studying. 6. As all shops were closed, we couldn't buy any peripheral for our computer. 7. As the book had been printed, we hoped to get it soon. 8. As the key had been lost, we couldn't get into the study. 9. After the computer had processed the data on the aircraft arrivals and departures, the plan of the airport's work was made. 10. After a new method had been studied in detail, the committee decided to introduce it at almost all the plants.

Exercise 29. *Translate into English using Objective or Absolute Participial Complex.*

1. Ми спостерігали, як вони грали в нову комп'ютерну гру. 2. Вони чули, як батьки розмовляли англійською мовою. 3. Ми бачили, як викладач заходив до аудиторії. 4. Оскільки лектор був хворий, у нас не було другої пари. 5. Ми помітили, як вона засміялася. 6. Мені відремонтували комп'ютер. 7. Коли моя сестра закінчила школу, вона вступила до університету. 8. Нам дали запрошення на виставку новітніх комп'ютерних технологій. 9. Оскільки було вже пізно, всі лабораторії були зачинені. 10. Я не зміг надрукувати статтю тому що принтер був несправний. 11. Коли заняття закінчилися, студенти нашої групи пішли додому.

Exercise 30. *Study the vocabulary to text 3.*

Department – відділ
engineering approach – технічний підхід
System Development Life Cycle (SDLC) – життєвий цикл розроблення системи
accomplish – здійснювати; досягати
outsource – 1) віддавати роботу стороннім, переводити виробництво в інший регіон; 2) залучати зовнішній ресурс
off-shoring – практика переміщення робочої бази компанії в іншу країну

land information system – ландшафтна інформаційна система
emerge – виникати, зароджуватись
spatial information system – просторова інформаційна система
recognition – 1) визнання; 2) у/впізнавання, розпізнавання
specification – 1) специфікація; 2) уточнення, конкретизування
maintenance – технічне обслуговування
concerned with – пов'язаний (*з чимось*)
core – основний, центральний
identity – 1) тотожність, ідентичність; 2) справжність, істинність; 3) індивідуальність, особа
be subject to smth – перебувати під впливом чогось, бути об'єктом чогось
debate – дебати, дискусія
scholar – вчений
artifact – артефакт
interplay – взаємодія

Exercise 31. *Translate the word combinations below into Ukrainian*.

Information technology department; to strongly influence; a series of methodologies; in order to develop and use an information system; engineering approach; a systematic procedure; to occur in sequence; can be accomplished; the geographical distribution; can be broadly considered; information gathering; system design; system implementation; subject to debate among scholars.

Exercise 32. *Read and translate text 3.*

Text 3. Information System Development

Information technology department in larger organizations tends to strongly influence information technology development, use and application in the organizations, which may be a business or corporation. A series of methodologies and processes can be used in order to develop and use an information system. Many developers have used a more engineering approach such as the System Development Life Cycle (SDLC) which is a systematic procedure of developing an information system through stages that occur in sequence. An information system can be developed in house (within the organization). This can be accomplished by outsourcing certain components or the entire system. A specific case is the geographical distribution of the development team (Off-shoring, Global information system).

Geographic information systems, land information systems and disaster information systems are also some of the emerging information systems but they can be broadly considered as spatial information systems. System development is done in stages which include: problem recognition and specification, information gathering, requirements specification for the new system, system design, system construction, system implementation, review and maintenance, information systems research, which is generally concerned with the study of the effects of information systems on the behaviour of individuals, groups and organizations.

Although information systems as a discipline has been evolving for over 30 years now, the core focus or identity of IS research is still subject to debate among scholars. There are two main views around this debate: a narrow view focusing on the IT artifact as the core subject matter of IS research, and a broad view that focuses on the interplay between social and technical aspects of IT.

Exercise 33. *Find in text 3 the English for:*

застосування в організаціях; розроблення інформаційної системи; поетапно; інформаційні системи, що з'являються; визначення та уточнення завдання; визначення вимог до нової системи; дослідження інформаційних систем; дослідження впливів інформаційних систем; поведінка людей, груп та організацій; основна увага; два головні погляди; взаємодія між соціальними та технічними аспектами інформаційних технологій.

Exercise 34. *Make up questions to the italicized parts of the sentences.*

1. ***A series of methodologies and processes*** can be used ***in order to develop and use an information system***. 2. An information system can be developed ***in house***. 3. ***System development*** is done ***in stages***. 4. Information system research is generally concerned ***with the study of the effects of information systems on the behaviour of individuals***. 5. There are **two** main views around this debate. 6. A broad view focuses on the interplay between **social and technical aspects of IT**.

Exercise 35. *Do the following assignments.*

1. Explain a distinction between information systems, computer systems and business processes.

2. Compare various types of information systems.
3. Name a series of methodologies used for an information system development.

Exercise 36. *Study the vocabulary to text 4.*

Executive information system (EIS) – інформаційна система для керівників
intend – призначати
senior executive – керівник вищого рангу
chief executive officer – керівник, головна посадова особа, генеральний директор *(компанії, підприємства)*
emphasis – акцент, наголос, особлива увага
drill-down capabilities – можливості деталізування
enterprise-wide – (загально)корпоративний, у масштабі підприємства
highlight – підкреслювати, зосереджувати увагу (*на чомусь*), наголошувати (*щось*)
chief executive officer – керівник, головна посадова особа, генеральний директор *(компанії, підприємства)*
objective – мета
cross – перетинати
adopt – приймати, запроваджувати
arrangement – організація
customize – налаштовувати, адаптувати

Exercise 37. *Translate the word combinations below into Ukrainian*.

Intended to facilitate and support the information; information relevant to meeting the strategic goals of the organization; easy-to-user interfaces; to help top-level executives analyze, compare, and highlight trends; mainframe computer-based programs; to provide sales performance; chief executive officer; to satisfy senior executives' needs; cross computer hardware platforms; to get access to the company's data.

Exercise 38. *Read and translate text 4.*

Text 4. An Executive Information System

An Executive Information System (EIS) is a type of management information system intended to facilitate and support the information and decision-making needs of senior executives by providing easy

access to both internal and external information relevant to meeting the strategic goals of the organization. It is commonly considered as a specialized form of a Decision Support System (DSS).

The emphasis of EIS is on graphical displays and easy-to-user interfaces. They offer strong reporting and drill-down capabilities. In general, EIS is enterprise-wide DSS that helps top-level executives analyze, compare, and highlight trends in important variables so that they can monitor performance and identify opportunities and problems.

Traditionally, executive information systems were developed as mainframe computer-based programs. The purpose was to package a company's data and to provide sales performance or market research statistics for decision makers, such as financial officers, marketing directors, and chief executive officers. The objective was to develop computer applications that would highlight information to satisfy senior executives' needs.

Today the application of EIS is not only in typical corporate hierarchies, but also at personal computers on a local area network. EIS now crosses computer hardware platforms and integrates information stored on mainframes, personal computer systems, and minicomputers. As some client service companies adopt the latest enterprise information systems, employees can use their personal computers to get access to the company's data and decide which data are relevant for their decision makings. This arrangement makes all users able to customize their access to the proper company's data and provide relevant information to both upper and lower levels in companies.

Exercise 39. *Find in text 4 the English for:*

інформаційна система для керівників; шляхом забезпечення легкого доступу; зазвичай розглядати як; графічні зображення; можливості деталізації; визначати перспективи та проблеми; статистика дослідження ринку; локальна мережа; використовувати свої персональні комп'ютери для отримання доступу до бази даних компанії; вирішувати, яка інформація є суттєвою для прийняття рішень.

Exercise 40. *Answer the questions on text 4.*

1. What is an executive information system? 2. What is it commonly considered as? 3. What does EIS emphasize? 4. What does EIS help top-level executives do? 5. What were executive information systems

developed as? What for? 6. What is the application of EIS today? 7. What enables employees to get an access to the company's data using their personal computers? 8. What does this arrangement allow?

Exercise 41. *Fill in the blanks with prepositions* ***in, on, of, to, at, by, as, for*** *where necessary.*

1. EIS is intended to facilitate the information and decision-making needs ... senior executives ... providing an access ... internal and external information relevant ... meeting the strategic goals ... the organization. 2. The application of EIS is also ... personal computers ... a local area network. 3. The purpose is to provide statistics ... decision makers. 4. ... general, EIS helps top-level executives analyze trends ... important variables. 5. It is considered ... a specialized form ... DSS. 6. The emphasis is ... graphical displays and easy-to-user interfaces. 7. EIS was developed ... mainframe computer-based programs. 8. EIS integrates information stored ... mainframes, PCs and minicomputers. 9. Employees can use their PCs to get access ... the company's data relevant ... their decision makings. 10. All users are able to customize their access ... the proper company's data and provide relevant information ... upper and lower levels ... companies.

Exercise 42. *Decipher the abbreviations below:*
IS, IT, EIS, DSS, ICT, SDLC, PC.

Exercise 43. *Find some additional information and speak on:*

1. Information System Concept.
2. Information System Types.
3. Information System Development.

UNIT 11. COMPUTER SECURITY

Exercise 1. *Study the basic vocabulary.*

a) terms
computer security – комп'ютерна безпека
web server – веб-сервер
computer technology – обчислювальна техніка
(un)authorized use – (не)законне/(не)правочинне/(не)санкціоноване використання

(un)authorized user – (не)правочиний користувач
break (*p.*broke, *pp.* broken) into a system – «зламувати» систему (*незаконно входити до системи*)
protection of information – захист інформації
confidentiality – конфіденційність
integrity – цілісність *(даних)*
authentication – автентифікація, перевірка *(справжності)*
trustworthiness of data – достовірність даних
hacking – хакерство
compromise – 1) порушувати *(конфіденційність)*, розголошувати *(таємну інформацію)*; 2) зламувати *(систему)*
deny (*access/resources*) – відмовляти (*у доступі/ресурсах*); не давати доступитися до інформації; заперечувати
denial of service attack – атака типу «відмова обслуговування» *(дія, що спричиняє відмову обслуговування законних користувачів)*
access control – контроль доступу
maintain control – підтримувати контроль
nonrepudiation – неможливість заперечення авторства
privacy – конфіденційність / приватність персональної інформації
register with an Internet site – реєструватися на інтернет-сайті
sensitive information – таємна / конфіденційна інформація
(un)trustworthy – (не)надійний
computer crime – комп'ютерна злочинність, використання комп'ютера для скоєння правопорушень
prevention – запобігання
detection – виявлення
remediation – виправлення, *тут:* ліквідування наслідків
anonymity – анонімність

b) nouns
concern – 1) занепокоєння, турбота; 2) зацікавленість, інтерес; 3) участь
count – підрахунок
implication – підтекст, зміст; те, що мають на увазі
intruder – особа, що незаконно вдерлася; зламник
theft – крадіжка
corruption – 1) зміна, викривлення *(інформації, тексту)*; 2) пошкодження
tampering – 1) псування; 2) фальшування, підроблення

collapse – 1) руйнування; 2) вихід з ладу
party – сторона, суб'єкт, учасник
breach – порушення
disclosure – розкриття, викриття
composite – комплекс, сукупність
reach – 1) розмах; 2) сфера (*впливу*); 3) простір, протяжність; 4) радіус дії

c) verbs
hide (*p.* hid, *pp.* hidden) – ховати
attempt – пробувати, робити спробу, намагатися
remain – залишатися
phrase – висловлювати
claim – стверджувати, заявляти
entitle (to) – давати право (*на щось*)
assure – гарантувати; забезпечувати

d) adjectives
crippled – (не)придатний; бракований; зіпсований
sensitive – 1) чутливий; 2) конфіденційний
elusive – 1) невиразний, непевний; 2) важкодосяжний
embarrassing – такий, що заплутує (*когось*); ускладнює (*щось*)
disastrous – катастрофічний, згубний
(im)proper – (не)правильний, (не)належний, (не)відповідний
detectable – такий, що може бути виявлений, виявний
(un)available – (не)доступний
convinced – впевнений, переконаний

e) adverbs
constantly – постійно
respectively – відповідно
somewhat – певною мірою, почасти, дещо
instead (of) – замість
deliberately – навмисно, свідомо
legitimately – законно, обґрунтовано
primarily – головним чином; передусім
particularly – 1) особливо; 2) зокрема
conversely – навпаки
perhaps – можливо

f) conjunctions
while – в той час як; тоді як
if/whether – чи

Exercise 2. *Choose verbs among the following words. Put the first letters of the verbs into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Reliability, properly, collect, primarily, modification, occur, legitimately, maintain, accurate, theft, provide, deliberately, particular, remain, accessible, organize, remediation, detectable, motivate, untrustworthy, prevention, imply, embarrassing, store, detectable, enable.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) security, worry, concern, significant, computation, shrink, common, accurate, automobile, determine, attempt, objective, include, collapse, untrustworthy, individual, major, breach, restrict, improper, unauthorized, desired, deliberately, primarily, hide, in addition to;

b) security, significant, strong, early, military, easy, common, accurate, include, accessible, intended, valuable, often, instead of, major, restrict, relevant, deliberately, deny, broad.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: differ – difference – differentiator – different – differently

Differ, protect, access, communicate, change, mean, process, deny, industry, restrict, system, relevant, expect, imply, receive, apply, able, maintain, collect.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

To stop unauthorized users from accessing any part of your computer system; the objective of computer security; to protect from publication, tampering or collapse; the strategies and methodologies of computer security; elusive objective; enabling wanted computer behaviour; breaches of confidentiality; the need for keeping information secret; authorized parties; "denial of service" attacks; attacks against availability;

confidentiality applied to an individual; to register with an Internet site; a person's information seeking habits; to maintain control over what information is collected, how it is used, who may use it, and what purpose it is used for; the conceptual reach of computer security; prevention, detection, and remediation of attacks; identity and anonymity in cyberspace.

Exercise 6. *Read and translate text 1.*

Text 1. Computer Security Strategies and Methodologies

People have worried about the security of their computers for many years, and computer security concerns have always been a significant factor in the development and application of computer technology throughout society.

In the early development of computers, security was a strong factor – the computations that motivated their development had military applications. But the computers themselves were so big and so few that they were relatively easy to protect simply by limiting physical access to them, to their programmers and operators. Today, computers have shrunk so that a web server can be hidden in a matchbox and have become so common that few people can give an accurate count of the number they have in their homes and automobiles. Computers constantly communicate with one another; an isolated computer is crippled. The meaning and implications of "computer security" have changed over the years as well.

Today computer security refers to the branch of computer technology that deals with preventing and detecting unauthorized use of your computer. Prevention measures help you to stop unauthorized users (also known as "intruders") from accessing any part of your computer system. Detection helps you to determine whether or not someone attempted to break into your system, if they were successful, and what they may have done.

The objective of computer security includes protection of information and property from theft, corruption, or natural disaster, while allowing the information and property to remain accessible and productive to its intended users. The term computer system security means the collective processes and mechanisms by which sensitive and valuable information and services are protected from publication, tampering or col-

lapse by unauthorized activities or untrustworthy individuals and unplanned events respectively. The strategies and methodologies of computer security often differ from most other computer technologies because of its somewhat elusive objective of preventing unwanted computer behaviour instead of enabling wanted computer behaviour.

The major technical areas of computer security are usually represented by the initials CIA: *confidentiality, integrity*, and *availability* or *authentication*.

Confidentiality means that information cannot be accessed by unauthorized parties. Confidentiality is also known as secrecy; breaches of confidentiality range from embarrassing to disastrous. The need for keeping information secret arises from the use of computers in sensitive fields such as government and industry. For example, military and civilian institutions in the government often restrict access to information to those who need that information.

Integrity refers to the trustworthiness of data or resources, and it is usually phrased in terms of preventing improper or unauthorized change. Integrity means that information is protected against unauthorized changes that are not detectable to authorized users; many incidents of hacking compromise the integrity of databases and other resources.

Availability refers to the ability to use the information or resource desired. Availability is an important aspect of reliability as well as of system design because an unavailable system is at least as bad as no system at all. The aspect of availability that is relevant to security is that someone may deliberately arrange to deny access to data or to a service by making it unavailable. Availability means that resources are accessible by authorized parties; "denial of service" attacks, which are sometimes the topic of national news, are attacks against availability.

Authentication means that users are who they claim to be. Authentication is really a prerequisite for the first three properties, since without proper authentication it is not possible to determine whether a disclosure or modification has been properly authorized.

Other important concerns of computer security professionals are *access control* and *nonrepudiation*. Maintaining **access control** means not only that users can access only those resources and services to which they are entitled, but also that they are not denied resources that they legitimately can expect to access. **Nonrepudiation** refers to assuring that a neutral third party can be convinced that a particular transaction or

event did (or did not) occur. It is primarily of interest in the context of communication protocols, particularly for legal or financial transactions. It implies that a person who sends a message cannot deny that he sent it and, conversely, that a person who has received a message cannot deny that he received it.

While confidentiality, integrity, availability, authenticity, access control, and nonrepudiation are the most important concerns of a computer security manager, *privacy*, which can be viewed as confidentiality applied to an individual, is perhaps the most important aspect of computer security for everyday Internet users. Although users may feel that they have nothing to hide when they are registering with an Internet site or service, privacy on the Internet is about protecting one's personal information, even if the information does not seem sensitive. Because of the ease with which information in electronic format can be shared among companies, and because small pieces of related information from different sources can be easily linked together to form a composite of, for example, a person's information seeking habits, it is now very important that individuals are able to maintain control over what information is collected about them, how it is used, who may use it, and what purpose it is used for.

In addition to these technical aspects, the conceptual reach of computer security is broad and multifaceted. Computer security is concerned with topics such as computer crime; the prevention, detection, and remediation of attacks; and identity and anonymity in cyberspace.

Exercise 7. *Find in text 1 the English for:*

недоступна система; щоденні користувачі Інтернету; розвиток і застосування комп'ютерних технологій в усьому суспільстві; небажана поведінка комп'ютера; незаконне використання; залишатися доступним і корисним; незаплановані події; порушувати цілісність баз даних; концептуальний простір комп'ютерної безпеки; відповідна інформація з різних джерел; сукупність процесів і механізмів; обмежувати доступ до інформації; неправочинний користувач; пов'язаний із секретністю галузі; захист інформації та власності від викрадення, пошкодження або природного лиха; ненадійні особи; неправочинні суб'єкти; визначити, чи намагався хтось «зламати» вашу систему; військові та цивільні установи; точно підраховувати; на ранній стадії розвитку комп'ютерів; обрані користу-

вачі; питання комп'ютерної безпеки; обмежувати фізичний доступ; запобіжні заходи; непідімкнений до мережі комп'ютер; юридичні чи фінансові транзакції; незаконна діяльність; достовірність даних або ресурсів; забезпечення контролю доступу; навмисно влаштовувати заборону доступу до даних; конфіденційна та цінна інформація; запобігання недоречним або несанкціонованим змінам.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. Only recently have computer security concerns become a significant factor in the development and application of computer technology throughout society. 2. Earlier computers were relatively easy to protect by using specially designed software. 3. In the early development of computers the computations that motivated their development had civilian applications. 4. Computers constantly communicate with one another; an isolated computer is crippled. 5. The meaning and implications of "computer security" have never changed. 6. Today computer security refers to the branch of computer architecture that deals with protecting computer networks from malicious intrusion. 7. Confidentiality means that information cannot be accessed by intended users. 8. Breaches of confidentiality range from embarrassing to disastrous. 9. The need for keeping information secret arises from the use of computers in such fields such sports and tourism. 10. Integrity refers to the trustworthiness of data or resources, and it is usually phrased in terms of preventing necessary change. 11. Authentication refers to the ability to use the information or resource desired. 12. Availability means that resources are accessible by authorized parties. 13. Access control implies that a person who sends a message cannot deny that he sent it and, conversely, that a person who has received a message cannot deny that he received it. 14. Privacy can be viewed as confidentiality applied to an individual. 15. The conceptual reach of computer security is narrow and highly specialized.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to worry about	a) trustworthiness of data or resources
2) to motivate	b) confidentiality

3) to protect by	c) other computer technologies
4) to differ from	d) the security of computers
5) to range from	e) limiting access
6) to restrict	f) control
7) to refer to	g) among companies
8) to be viewed as	h) development
9) to be shared	i) access to information
10) to maintain	j) embarrassing to disastrous

Exercise 10. *Fill in the blanks with prepositions* ***on, in, to, of, by, with, about, for, throughout, from, against*** *where necessary.*

1. People have worried … the security … their computers … many years. 2. Computer security concerns have always been a significant factor … the development and application … computer technology … society. 3. Earlier computers were relatively easy to protect simply … limiting physical access … them, … their programmers and operators. 4. Today computer security refers … the branch … computer technology that deals … preventing and detecting unauthorized use … your computer. 5. Prevention measures help you to stop unauthorized users … accessing … any part … your computer system. 6. Breaches … confidentiality range … embarrassing … disastrous. 7. The need … keeping information secret arises … the use … computers … sensitive fields such as government and industry. 8. Integrity means that information is protected … unauthorized changes that are not detectable … authorized users. 9. The aspect … availability that is relevant … security is that someone may deliberately arrange to deny … access … data or … a service … making it unavailable. 10. Privacy, which can be viewed as confidentiality applied … an individual, is perhaps the most important aspect … computer security … everyday Internet users. 11. Privacy … the Internet is … protecting one's personal information, even if the information does not seem sensitive. 12. … addition … these technical aspects, the conceptual reach … computer security is broad and multifaceted.

Exercise 11. *Fill in the blanks with proper terms (****access control, computer security, authentication, availability, prevention measures, non-repudiation, privacy, confidentiality****) to complete the sentences.*

1. _____ is the most important aspect of computer security for everyday Internet users that can be viewed as confidentiality applied to an

individual. 2. _____ is assuring that a neutral third party can be convinced that a particular transaction or event did (or did not) occur. 3. _____is the ability to use the information or resource desired. 4. _____ is the branch of computer technology that deals with preventing and detecting unauthorized use of your computer. 5. _____is the property meaning that information cannot be accessed by unauthorized parties. 6. _____ is assuring that users are who they claim to be. 7. _____ is allowing users to access only those resources and services to which they are entitled and not denying resources that they legitimately can expect to access. 8. _____are measures that help you to stop unauthorized users (also known as "intruders") from accessing any part of your computer system.

Exercise 12. *Answer the questions on text 1.*

1. Was computer security a strong factor in the early development of computers? Why? 2. Why were earlier computers easy to protect? 3. What is computer security today? 4. What are the purposes of prevention and detection measures in terms of computer security? 5. What does the objective of computer security include? 6. Why do the strategies and methodologies of computer security differ from most other computer technologies? 7. What are the major technical areas of computer security? 8. What does confidentiality of information imply? 9. What does the need for keeping information secret arise from? 10. What does integrity of information refer to? 11. What is availability of information? 12. What does authentication mean? 13. What does maintaining access control mean? 14. What is nonrepudiation? 15. What is privacy on the Internet? Why is it important? 16. How can the conceptual reach of computer security be described? What topics is it concerned with?

Exercise 13. *Put all possible questions to the sentences below.*

1. Computers constantly communicate with one another. 2. Detection helps you to determine whether or not someone attempted to break into your system. 3. The objective of computer security includes protection of information and property from theft, corruption, or natural disaster. 4. The strategies and methodologies of computer security often differ from most other computer technologies. 5. Breaches of confidentiality

range from embarrassing to disastrous. 6. Availability refers to the ability to use the information or resource desired. 7. Many incidents of hacking compromise the integrity of databases and other resources.

Exercise 14. *Translate into English.*

1. Питання комп'ютерної безпеки завжди було важливим чинником розроблення та застосування комп'ютерних технологій в усьому суспільстві. 2. На ранній стадії розвитку комп'ютерів безпека була важливим чинником захисту, оскільки обчислення, що слугували причиною їх розроблення, мали військове застосування. 3. Але самі комп'ютери були такими великими за розмірами, але такими нечисленними, що забезпечувати їх захист було відносно легко, лише обмеживши фізичний доступ програмістам та операторам до них. 4. Сьогодні комп'ютерною безпекою називають галузь обчислювальної техніки, предметом якої є запобігання незаконному використанню вашого комп'ютера та виявлення цього. 5. Система виявлення допомагає визначити, чи намагався хтось «зламати» вашу систему, чи вдалося це йому і що він міг зробити. 6. Комп'ютерна безпека призначена для захисту інформації та власності від викрадення, пошкодження або природного лиха, водночас дає можливість залишати інформацію та власність доступними та корисними для обраних користувачів. 7. Термін «комп'ютерна безпека» означає сукупність процесів і механізмів, з допомогою яких конфіденційна та цінна інформація і послуги захищаються від оприлюднення, псування та знищення ненадійними особами або внаслідок незаконної діяльності чи незапланованих подій. 8. Конфіденційність означає, що до інформації не можуть доступитися неправочинні суб'єкти. 9. Необхідність тримати інформацію в таємниці виникає через використання комп'ютерів у пов'язаних із секретністю галузях, таких як державне управління та промисловість. 10. У військових та цивільних урядових установах доступ до інформації часто обмежують до кола осіб, яким ця інформація необхідна. 11. Цілісністю називають достовірність даних або ресурсів, і зазвичай про неї говорять в контексті запобігання недоречним або несанкціонованим змінам. 12. Багато випадків хакерства порушують цілісність баз даних та інших ресурсів. 13. Доступністю називають можливість користуватися потрібною інформацією або ресурсом. 14. Атаки типу «відмова обслуговування», які іноді є темою загальнонаціональних новин, є нападами на доступність. 15. Автентифікація означає, що корис-

тувач справді є тим, ким він стверджує. 16. Без належної автентифікації неможливо визначити, чи правочинними є внесені зміни. 17. Забезпечення контролю доступу означає не тільки те, що користувачі можуть мати доступ лише до тих ресурсів і послуг, на які вони мають право, але також і те, що їм не заборонено користуватися ресурсами, до яких вони обґрунтовано сподіваються доступитися. 18. Неможливість заперечення авторства означає, що особа, яка надсилає повідомлення, не може заперечувати, що вона його надіслала, і, навпаки, що особа, яка отримала повідомлення, не може заперечувати, що вона його отримала. 19. Приватність, яка може розглядатися як конфіденційність стосовно окремої особи, можливо є найважливішим аспектом комп'ютерної безпеки для щоденних користувачів Інтернету. 20. Приватність в Інтернеті – це захист особистої інформації, навіть якщо ця інформація не видається конфіденційною. 21. Зараз дуже важливо, щоб людина могла контролювати, яку інформацію про неї збирають, як, хто і з якою метою її використовує.

Exercise 15. *Retell the text "Computer Security Strategies and Methodologies".*

Exercise 16. *Study the vocabulary to text 2.*

Vulnerability – слабке місце, вразливість, чутливість (*що*)*до чогось*)
secure – 1) безпечний; 2) охороняти, захищати, забезпечувати, гарантувати
threat – загроза, небезпека
exploit – програма атаки / зламу
eavesdropping – підслуховування, перехоплення
social engineering – «соціальна інженерія», соціотехніка *(тактика обдурювання користувачів мережі чи адміністратора, що її використовують зловмисники з метою отримання паролів, необхідних для проникнення у захищену систему)*
human error – суб'єктивна помилка, помилка людини
indirect attack – непряма / опосередкована атака
backdoor – лазівка, таємний вхід, чорний вхід, шлях обходу системи захисту
direct access attack – атака прямого доступу
chunk (*of data*) – порція (*даних*)

glitch – програмна помилка, *розм.* «глюк»
privilege escalation – розширення привілеїв
flaw – дефект
remote – віддалений, дистанційний
media file – мультимедійний файл, медіа файл
surreptitious – таємний
faint – слабкий
penetrate – проникати
carelessness – неуважність, необережність, легковажність, недбальство, недотримання правил безпеки
deceive – обманювати
deception – обман, брехня
gain – здобувати, діставати, отримувати
render – (*у сполученні з прикметником*) спричиняти певний стан, робити
consecutive – послідовний
botnet – ботнет *(мережа комп'ютерів, уражених програмою, яка підтримує зв'язок з її розробниками для того, щоб надсилати пошту без запиту, атакувати вебсайти)*
flood – 1) повінь, потік непотрібної інформації, *розм.* «флуд»; 2) наповнювати, затоплювати
target system – цільова система
exhaust – вичерпувати
attack amplifier – підсилювач атак
File Transfer Protocol (FTP) протокол передання файлів, протокол FTP
DNS – 1) Domain Name System – служба імен доменів; 2) Domain Name Server – сервер доменних імен
launch – запускати, починати
commonly – зазвичай, звичайно, переважно
malfunction – 1) несправна робота, неправильне функціонування; 2) неправильно працювати, не спрацьовувати
crash –1) аварійна відмова, збій; 2) зазнати аварії / збою
anonymize – робити анонімним, приховувати особисту інформацію
cryptosystem – криптосистема
bypass – 1) обхід; 2) обходити
remote access – віддалений доступ
plain text – незашифрований / незакодований текст
remain undetected – залишатися невиявленим

rootkit – *руткіт (програмні засоби, що приховують наслідки зламу та ховають засоби, які використовують зловмисники, від антивірусного програмного забезпечення)*
binary – двійковий файл
hook (*into*) – підмикатися (*до чогось*)
fake – підроблювати, дурити
worm – черв'як (*програма, що самостійно поширює свої копії мережею*)
keylogger – логер клавіатури *(програма чи апаратний пристрій, що реєструє кожне натискання клавіш на клавіатурі комп'ютера)*
covert – таємний, невидимий, секретний
backup media – резервні носії
key drive – флеш-пам'ять
boot – завантажувати, виконувати початкове завантаження
bootable media – завантажувальний носій
defeat – перемагати, ліквідовувати

Exercise 17. *Translate the word combinations below into Ukrainian.*

The techniques for securing a computer system; exploits, eavesdropping, social engineering and human error; denial of service attacks, indirect attacks, backdoors and direct access attacks; a chunk of data; to take advantage of a bug, glitch or vulnerability; in order to cause unintended or unanticipated behaviour; to occur on computer software, hardware, or something electronic; gaining control of a computer system; allowing privilege escalation or a denial of service attack; to ensure the quality of any code released; to fail to discover extremely unusual potential exploits; to be frequently reused in Trojan horses and computer viruses; the act of surreptitious listening to a private conversation; to penetrate the system; malicious individuals; to deliberately deceive the individuals; a large number of compromised hosts; to flood a target system with network requests, to render the system unusable through resource exhaustion; attack amplifier; to launch the flood; common vulnerabilities in applications; merely make the target application malfunction or crash; a method of bypassing normal authentication; remote access to a computer; to obtain access to plain text; to remain undetected; rootkits, which hook into the function calls of the operating system; to fake information about disk and

memory usage; common consumer devices; to compromise security; software worms, keyloggers, and covert listening devices; to download large quantities of data onto backup media; to defeat the exploit; to encrypt the storage media.

Exercise 18. *Read and translate text 2.*

Text 2. Vulnerabilities

To understand the techniques for securing a computer system, it is important to first understand the various types of "attacks" that can be made against it. The threats can typically be classified into these seven categories: exploits, eavesdropping, social engineering and human error, denial of service attacks, indirect attacks, backdoors, direct access attacks.

Exploits. An exploit (from the same word in the French language, meaning "achievement", or "accomplishment") is a piece of software, a chunk of data, or sequence of commands that take advantage of a bug, glitch or vulnerability in order to cause unintended or unanticipated behaviour to occur on computer software, hardware, or something electronic (usually computerized). This frequently includes such things as gaining control of a computer system or allowing privilege escalation or a denial of service attack. Many development methodologies rely on testing to ensure the quality of any code released; this process often fails to discover extremely unusual potential exploits. The term "exploit" generally refers to small programs designed to take advantage of a software flaw that has been discovered, either remote or local. The code from the exploit program is frequently reused in Trojan horses and computer viruses. In some cases, a vulnerability can lie in certain programs' processing of a specific file type, such as a non-executable media file.

Eavesdropping. Eavesdropping is the act of surreptitious listening to a private conversation. Even machines that operate as a closed system (i.e. with no contact to the outside world) can be eavesdropped upon via monitoring the faint electro-magnetic transmissions generated by the hardware. The FBI's[1] proposed Carnivore program was intended to act as a system of eavesdropping protocols built into the systems of Internet service providers.

[1]FBI – Federal Bureau of Investigation – Федеральне бюро розслідувань, ФБР (*США*)

Social engineering and human error. A computer system is no more secure than the human systems responsible for its operation. Malicious individuals have regularly penetrated well-designed, secure computer systems by taking advantage of the carelessness of trusted individuals, or by deliberately deceiving them, for example sending messages that they are the system administrator and asking for passwords. This deception is known as social engineering.

Denial of service attacks. Unlike other exploits, denial of service attacks are not used to gain unauthorized access or control of a system. They are instead designed to render it unusable. Attackers can deny service to individual victims, such as by deliberately guessing a wrong password 3 consecutive times and thus causing the victim account to be locked, or they may overload the capabilities of a machine or network and block all users at once. These types of attack are, in practice, very hard to prevent, because the behaviour of whole networks needs to be analyzed, not only the behaviour of small pieces of code. Distributed denial of service (DDoS) attacks are common, where a large number of compromised hosts (commonly referred to as "zombie computers", used as part of a botnet with, for example a worm, Trojan horse, or backdoor exploit to control them) are used to flood a target system with network requests, thus attempting to render it unusable through resource exhaustion. Another technique to exhaust victim resources is through the use of an attack amplifier – where the attacker takes advantage of poorly designed protocols on third party machines, such as FTP or DNS, in order to instruct these hosts to launch the flood. There are also common vulnerabilities in applications that cannot be used to take control over a computer, but merely make the target application malfunction or crash. This is known as a denial-of-service exploit.

Indirect attacks. An indirect attack is an attack launched by a third party computer. By using someone else's computer to launch an attack, it becomes far more difficult to track down the actual attacker. There have also been cases when attackers took advantage of public anonymizing systems.

Backdoors. A backdoor in a computer system (or cryptosystem or algorithm) is a method of bypassing normal authentication, securing remote access to a computer, obtaining access to plain text, and so on, while attempting to remain undetected. The backdoor may take the form of an installed program (e.g., Back Orifice), or could be a modification

to an existing program or hardware device. A specific form of backdoors are rootkits, which replace system binaries and/or hook into the function calls of the operating system to hide the presence of other programs, users, services and open ports. They may also fake information about disk and memory usage.

Direct access attacks. Common consumer devices can be used to transfer data surreptitiously. Someone who has gained access to a computer can install any type of devices to compromise security, including operating system modifications, software worms, keyloggers, and covert listening devices. The attacker can also easily download large quantities of data onto backup media, for instance CD-R/DVD-R, tape; or portable devices such as key drives, digital cameras or digital audio players. Another common technique is to boot an operating system contained on a CD-ROM or other bootable media and read the data from the hard drive(s) this way. The only way to defeat this is to encrypt the storage media and store the key separate from the system.

Exercise 19. *Find in text 2 the English for*:

методи захисту комп'ютерної системи; програма атаки; підслуховування, «соціальна інженерія» та помилка людини; непряма (опосередкована) атака; шлях обходу системи захисту та атака прямого доступу; порція даних; скористатися збоєм, програмною помилкою чи слабким місцем; спричиняти непередбачувану та неочікувану поведінку; дозволяти розширення привілеїв; атака типу «відмова обслуговування»; забезпечувати якість будь-якого виданого коду; невиконуваний медіа файл; таємне підслуховування приватних розмов; зловмисник; проникати в добре спроектовані та захищені системи; скористатися неуважністю довірених осіб; навмисно когось обманювати; отримувати несанкціонований доступ; три рази поспіль; зламаний хост; наповнювати цільову систему запитами по мережі; змушувати цільове прикладне програмне забезпечення неправильно працювати або зазнавати збою; публічні системи, що приховують інформацію про особу; метод уникнення автентифікації; забезпечувати віддалений доступ до комп'ютера; підроблювати інформацію про використання пам'яті; таємно передавати інформацію; порушувати безпеку; таємні пристрої підслуховування; завантажувальні носії; шифрувати носії інформації.

Exercise 20. *Fill in the blanks with proper terms (**direct access attack, backdoor, indirect attack, exploit, eavesdropping, denial of service exploit, social engineering**) to complete the sentences.*

1. _____ is the act of surreptitiously listening to a private conversation. 2. _____ refers to small programs designed to take advantage of a software flaw that has been discovered, either remote or local. 3. _____is a vulnerability in application that merely makes the target application malfunction or crash. 4. _____ is regular penetrating well-designed, secure computer systems by taking advantage of the carelessness of trusted individuals, or by deliberately deceiving them, for example sending messages that they are the system administrator and asking for passwords. 5. _____ is using common consumer devices to transfer data surreptitiously. 6. _____ is a method of bypassing normal authentication, securing remote access to a computer, obtaining access to plain text, and so on, while attempting to remain undetected. 7. _____ is an attack launched by a third party computer.

Exercise 21. *Answer the questions on text 2.*

1. Why is it important to understand the various types of "attacks" that can be made against a computer system? 2. How many categories can threats be classified into? 3. What is an exploit? 4. What is eavesdropping? 5. How can malicious individuals penetrate well-designed, secure computer systems? 6. What is the difference between denial of service attacks and other exploits? 7. How can attackers deny service to individual victims? 8. What techniques can malicious individuals use to exhaust the victim's resources? 9. What is an indirect attack? 10. What is a backdoor in a computer system? 11. What forms may the backdoor take? 12. What kind of devices can be installed to compromise computer security?

Exercise 22. *Speak on different computer system vulnerabilities.*

Exercise 23. *Study the vocabulary to text 3.*

Infectious – інфікований, заразний
malware – шкідливі програми
concealment – маскування

infiltrate – просочуватися, проникати
consent – згода, дозвіл
intrusive – набридливий, надокучливий
annoying – дратівний, набридливий, надокучливий
catch-all phrase – всеосяжна фраза
defective – недосконалий, дефектний
harmful – шкідливий
spyware – програмне забезпечення, призначене для шпигування за діями користувача
dishonest – нечесний, непорядний
adware – безкоштовний рекламний продукт; вірус, що скачує рекламу та спам
executable software – виконуване програмне забезпечення
payload – інформаційне наповнення
disguise – маскувати
innocuous – безпечний, нешкідливий
tempt – спокушати, зваблювати
immediately – негайно, невідкладно
dropper – скидач
outbreak – атака
inject – впорскувати, вводити, впускати
bundle – об'єднувати *(у пакет)*
in loose terms – у загальних рисах; невизначено
disinfection – знезараження, дезінфекція
repel – відбивати
casual inspection – випадкова, несподівана перевірка

Exercise 24. *Translate the word combinations below into Ukrainian.*

Infectious malware and concealment; to infiltrate a computer system without the owner's informed consent; hostile, intrusive, or annoying software or program code; a catch-all phrase; to include all types of malware; the intent of the creator; a legitimate purpose; to contain harmful bugs; Trojan horses; executable software; to carry a payload; to be classified as viruses rather than worms; to accomplish the goals; to be disguised as something innocuous or desirable; to be tempted to install; to start off a worm outbreak; by injecting the worm into users' local networks; to avoid detection and disinfection; to prevent a malicious process from being visible; to gain administrator access; to defend

against removal; bypassing normal authentication procedure; once a system has been compromised; to secure remote access to a computer, while attempting to remain hidden from casual inspection.

Exercise 25. *Read and translate text 3.*

Text 3. Malware and Concealment Methods

Malware, short for *malicious software*, is software designed to infiltrate a computer system without the owner's informed consent. The expression is a general term used by computer professionals to mean a variety of forms of hostile, intrusive, or annoying software or program code. The term "computer virus" is sometimes used as a catch-all phrase to include all types of malware, including true viruses.

Software is considered malware based on the intent of the creator rather than any particular features. Malware is not the same as defective software, that is, software that has a legitimate purpose but contains harmful bugs. Malware includes computer viruses, worms, Trojan horses, most rootkits, spyware, dishonest adware, and other malicious and unwanted software.

Infectious malware: viruses and worms. The best-known types of malware, *viruses* and *worms*, are known for the manner in which they spread, rather than any other particular behaviour. The term *computer virus* is used for a program that has infected some executable software and that causes that software, *when run*, to spread the virus to other executable software. Viruses may also contain a payload that performs other actions, often malicious. A *worm*, on the other hand, is a program that actively transmits itself over a network to infect other computers. It may carry a payload too.

These definitions lead to the observation that a virus requires user intervention to spread, whereas a worm spreads automatically. Using this distinction, infections transmitted by email or Microsoft Word documents, which rely on the recipient opening a file or email to infect the system, would be classified as viruses rather than worms.

Concealment: Trojan horses, rootkits, and backdoors.

Trojan horses. For a malicious program to accomplish its goals, it must be able to do so without being shut down, or deleted by the user or administrator of the computer on which it is running. Concealment can also help get the malware installed in the first place. When a malicious

program is disguised as something innocuous or desirable, users may be tempted to install it without knowing what it does. This is the technique of the *Trojan horse* or *Trojan*.

In broad terms, a Trojan horse is any program that invites the user to run it, concealing a harmful or malicious payload. The payload may take effect immediately and can lead to many undesirable effects, such as deleting the user's files or further installing malicious or undesirable software. Trojan horses known as droppers are used to start off a worm outbreak, by injecting the worm into users' local networks.

One of the most common ways that spyware is distributed is as a Trojan horse, bundled with a piece of desirable software that the user downloads from the Internet. When the user installs the software, the spyware is installed alongside. Spyware authors who attempt to act in a legal fashion may include an end-user license agreement that states the behaviour of the spyware in loose terms, which the users are unlikely to read or understand.

Rootkits. Once a malicious program is installed on a system, it is essential that it *stays* concealed, to avoid detection and disinfection. The same is true when a human attacker breaks into a computer directly. Techniques known as *rootkits* allow this concealment, by modifying the host operating system so that the malware is hidden from the user. Rootkits can prevent a malicious process from being visible in the system's list of processes, or keep its files from being read. Originally, a rootkit was a set of tools installed by a human attacker on a UNIX system where the attacker had gained administrator (root) access. Today, the term is used more generally for concealment routines in a malicious program.

Some malicious programs contain routines to defend against removal, not merely to hide themselves, but to repel attempts to remove them. Modern malware uses the techniques wherein the malware starts a number of processes that monitor and restore one another as needed.

Backdoors. A backdoor is a method of bypassing normal authentication procedures. Once a system has been compromised (by one of the above methods, or in some other way), one or more backdoors may be installed in order to allow easier access in the future. Backdoors may also be installed prior to malicious software, to allow attackers entry.

The idea has often been suggested that computer manufacturers preinstall backdoors on their systems to provide technical support for

customers, but this has never been reliably verified. Crackers typically use backdoors to secure remote access to a computer, while attempting to remain hidden from casual inspection. To install backdoors crackers may use Trojan horses, worms, or other methods.

Exercise 26. *Give derivatives of the words below and explain their meanings.*

Design, compute, system, express, profession, vary, intrude, create, defect, harm, honest, infect, behave, execute, act, define, rely, conceal, immediate, desire, direct, modify, origin, verify, install.

Exercise 27. *Write definitions of the terms below.*

Malicious software, defective software, a computer virus, a worm, a Trojan horse, spyware, a rootkit, a backdoor, a cracker.

Exercise 28. *Make key questions to the text "Malware and Concealment Methods".*

Exercise 29. *Write a summary of text 3.*

Exercise 30. *Speak on the ways you protect your computer from infectious malware.*

Exercise 31. *Memorize the forms of the Gerund.*

	Active	Passive
Present	buying	being bought
Perfect	having bought	having been bought

Exercise 32. *Translate the following sentences paying attention to the forms and functions of the Gerund. (It may be used as a subject, an attribute, an object, a predicative, an adverbial modifier).*

1. An information system activities are devoted to processing, capturing, transmitting, storing, retrieving, manipulating and displaying information. 2. He was sure of having answered correctly at the exam. 3. System analysts are interested in improving and updating some applications and specialized databases. 4. He insisted on being given a task. 5. Program designers are not only experts in writing particular

programs but they are also well experienced in hardware protecting. 6. Processing is handling and manipulating data for some purpose. 7. He was pleased with having won the competition. 8. Animation is the process of creating, usually graphically, a series of frames and then having them displayed rapidly to get a sense of movement. 9. Creating your own media is more interactive than using existing content, and collaborating with others in the creation of media is still more interactive. 10. He was proud of having been chosen the manager of the project. 11. Preparing a program begins with a complete description of the job to be done. 12. A program is prepared by formulating a task and expressing it in appropriate computer language. 13. He was surprised at having been praised at the meeting. 14. The hardware protects the operating system image and file system privileges from being tempered.

Exercise 33. *Write out all Gerunds from the text of Exercise 25. Identify their forms and functions.*

Exercise 34. *Complete the sentences using the proper form of the Gerund.*

1. (Minimize) the number of high-consequence targets is one of the principles of (develop) secure software. 2. He is sure of (solve) the problem correctly. 3. The program needs (debug). 4. He is sorry for (be) inattentive at the test. 5. There is not much thought that has to go into (design) a program that performs only one or two tasks. 6. (Break) the program into sub-tasks gives you the chance to think it through before (write) a single line of code. 7. He objected to (give) a bad mark. 8. BASIC is well suited for (write) relatively simple programs for personal computers. 9. Computer operates by (manipulate) the charges that represent the numbers 0 and 1. 10. Most Multi-Function Printers (MFP) include (print), (scan), and (copy) among their features. 11. He is proud of (hear) his name in the list of best students. 12. He insisted on (give) this task by the teacher. 13. A printer is a peripheral used for (produce) a hard copy of documents stored in electronic form. 14. Nodes communicate across the network by (pass) data, address and token through the hub. 15. Validate every assumption made by the software or about the software before (act) on that assumption. 16. She was pleased by (win) this award. 17. Programmers must be cautious about (avoid) coding errors and (find) and (remove) the bugs they were unable to avoid.

Exercise 35. *Translate the sentences paying attention to the Gerundial Complexes.*

1. Parents insisted on their son entering the university. 2. Mother was proud of her daughter having passed the examination well. 3. Your identifying a small number of key patterns is sufficient for implementation of the system. 4. I like the computer operating properly. 5. I want her being sent to the conference. 6. Father dreams of his son becoming a software engineer. 7. I don't mind the computer being switched on. 8. The teacher was surprised at the student not knowing the lesson. 9. Parents were pleased with their daughter getting an excellent mark. 10. Your correcting the errors is very desirable. 11. The developer didn't have much hope of the program working. 12. Don't fear my forgetting the password. 13. I insist on our team solving this problem. 14. I insist on this software being maintained to discover its weaknesses and vulnerabilities. 15. Your obscuring specific data within a database table or cell is necessary to provide sensitive information being not exposed to unauthorized personnel. 16. Your erasing data ensures no private information being leaked when an asset is retired or reused. 17. The developer insisted on their partitioning the program during detailed design.

Exercise 36. *Paraphrase the sentences using Gerundial Complexes after the models.*

Model 1:	He has failed the exam. We are surprised at it. – We are surprised at **his having failed the exam**.
Model 2:	The teacher wanted the students to do this laboratory work. – The teacher insisted on **the students doing this laboratory work.**

1. He misses classes. I dislike it. 2. You know so many programming languages, I am surprised at it. 3. You should go to the conference. I insist on it. 4. I want these students to take part in the competition. 5. He works as a software engineer. His parents are proud of him. 6. The teacher wanted the students to learn the rule. 7. The lecturer wanted the students to listen attentively and to take notes. 8. This software will work correctly. I am sure of it. 9. This program must be tested at once. I insist on it. 10. Bill Gates started Microsoft. Everybody knows it. 11.The results of the experiment must be explained. I insist on it. 12. He

has finished the experiment in time. I was told about it. 13. This problem was solved very quickly. Everybody was surprised at it. 14. The main objective of computer security is to protect information from theft, corruption and natural disaster. Everybody knows it. 15. The architectural specification should characterize interface mechanisms so that infrastructure developers will know what software to build. The project manager insists on it.

Exercise 37. *Study the vocabulary to text 4.*

Profit – 1) прибуток, зиск, вигода; 2) давати прибуток
dialer – пристрій набору номерів
prank – жарти, пустощі, витівки
monetize – перетворювати на гроші
revenue – дохід, прибуток, виручка
pop-up ad – спливаюче вікно *(наприклад, в інтернет-рекламі)*
alter – змінювати
benefit – 1) ко́ристь, вигода, прибуток; 2) мати прибуток
advertisement – реклама, оголошення
affiliate – філіал
prosecution – судове переслідування
proxy – сервер-посередник, proxy-сервер
spammer – спамер (*той, хто розсилає спам*)
log in – входити в систему, реєструватися
Internet Relay Chat – система діалогового спілкування Інтернетом
upgrade – модернізувати, покращувати
resistant – стійкий, міцний
fraud – шахрайство
virtual item – віртуальний елемент
dial-up modem – модем комутованої лінії пересилання
toll – плата *(за послуги)*
toll call – міжміська телефонна розмова
premium-rate telephone number – телефонний номер привілейованого тарифу, номер з premium-тарифом

Exercise 38. *Read and translate text 4.*

Text 4. Financially-Motivated Malware

During the 1980s and 1990s, it was usually taken for granted that malicious programs were created as a form of vandalism or prank. More

recently, the greater share of malware programs have been written with a financial or profit motive in mind. This can be taken as the malware authors' choice to monetize their control over infected systems: to turn that control into a source of revenue.

Spyware programs are commercially produced for the purpose of gathering information about computer users, showing them pop-up ads, or altering web-browser behaviour for the financial benefit of the spyware creator. For instance, some spyware programs redirect search engine results to paid advertisements. Others overwrite affiliate marketing codes so that revenue is redirected to the spyware creator rather than the intended recipient.

Spyware programs are sometimes installed as Trojan horses of one sort or another. They differ in that their creators present themselves openly as businesses, for instance by selling advertising space on the pop-ups created by the malware. Most such programs present the user with an end-user license agreement that may protect the creator from prosecution under computer laws.

Another way that financially-motivated malware creators can profit from their infections is to directly use the infected computers to do work for the creator. The infected computers are used as proxies to send out spam messages. A computer left in this state is often known as a zombie computer. The advantage to spammers of using infected computers is providing anonymity, protecting the spammer from prosecution. Spammers have also used infected PCs to target anti-spam organizations with distributed denial-of-service attacks.

In order to coordinate the activity of many infected computers, attackers have used coordinating systems known as *botnets*. In a botnet, the malware logs in to an Internet Relay Chat channel or other chat system. The attacker can then give instructions to all the infected systems simultaneously. Botnets can also be used to push upgraded malware to the infected systems, keeping them resistant to antivirus software or other security measures.

It is possible for a malware creator to profit by stealing sensitive information from a victim. Some malware programs install a *key logger*, which intercepts the user's keystrokes when entering a password, credit card number, or other information that may be exploited. This is then transmitted to the malware creator automatically, enabling credit card fraud and other theft. Similarly, malware may copy the CD key or

password for online games, allowing the creator to steal accounts or virtual items.

Another way of stealing money from the infected PC owner is to take control of a dial-up modem and dial an expensive toll call. *Dialer* software dials up a premium-rate telephone number such as a U.S. "900 number" and leaves the line open, charging the toll to the infected user.

Exercise 39. *Write the verbs related to the words below. Translate them.*

Different, presentation, denial, distribution, service, logger, dialer, behaviour, advertisement, agreement, prosecution, infection, directly, spammer, resistant, recipient, information, motivation, financial, choice, money, intent, activity, logger, transmission, owner.

Exercise 40. *Answer the questions on text 4.*

1. Has the greater share of malware programs been written as a form of vandalism or prank recently? 2. What are spyware programs produced for? 3. What are spyware programs sometimes installed as? 4. How can financially-motivated malware creators profit from their infections? 5. What is a zombie computer? 6. What systems are used by attackers in order to coordinate the activity of many infected computers? 7. What are botnets used for? 8. What is the principle of keylogger operation? 9. What is another way of stealing money from the infected PC owner?

Exercise 41. *Make questions to the italicized parts of the sentences.*

1. ***Malicious*** programs were created as a form of ***vandalism or prank***. 2. Spyware programs are commercially produced for the purpose ***of gathering information about computer users***. 3. Some ***spyware*** programs redirect search engine results ***to paid advertisements***. 4. Other programs overwrite ***affiliate marketing codes***. 5. They differ in ***that their creators present themselves openly as businesses***. 6. Another way that financially-motivated malware creators can profit from their infections is ***to directly use the infected computers to do work for the creator***. 7. The ***infected*** computers are used as proxies to send out ***spam messages***. 8. ***Spammers*** have also used ***infected PCs to target anti-spam organizations with distributed denial-of-service attacks***. 9. The attacker can then ***give instructions*** to all ***the infected systems***

simultaneously. 10. Dialer software dials up a ***premium-rate telephone*** number such as a U.S. "900 number" and leaves the line ***open***, charging the toll ***to the infected user***.

Exercise 42. *Write a summary of the text "Financially-Motivated Malware".*

Exercise 43. *Study the vocabulary to text 5.*

Data-stealing malware – шкідливі програми, що крадуть дані / інформацію
divest – позбавляти *(прав, повноважень, власності)*
proprietary information – службова інформація; інформація, що є власністю фірми, організації; конфіденційна інформація *(фірми, організації)*
underground – нелегальний, секретний, підпільний
content security threat – загроза безпеці контенту
bot (*скор. від* robot) – мережевий агент-робот *(програма, що автономно вирішує завдання)*
phishing – фішинг *(різновид Інтернет-шахрайства – випитування конфіденційної інформації з допомогою запитів, що мають вигляд офіційних листів)*
poisoning – *тут:* зміна, псування, викривлення
search engine optimization/optimizer (SEO) – оптимізація / оптимізатор пошукових систем
trace – слід, ознака
routinely – щодня, регулярно, як заведено
flush – 1) очищувати *(наприклад, вміст буферів оперативної пам'яті від старих даних)*; 2) скидати *(вміст кеша або буфера на диск)*
drive-by download process – процес автоматичного завантаження непотрібної програми в комп'ютер
host – містити
rogue – некерований
thwart – заважати, перешкоджати
Intrusion Detection System (IDS) – система виявлення (*мережевих*) атак
perceivable – відчутний
anomaly – відхилення від норми, аномалія
stealthy – непомітний, таємний, прихований
decryption – декодування, дешифрування

screenshot – моментальний знімок екрану
Data Loss Prevention (DLP) – попередження втрати даних
leakage protection – захист від втрати
hinge on sth – залежати (*від чогось*), *розм.* упиратися (*у щось*)
metadata – метадані (*дані з описом інших даних*)
tag – маркувати, розставляти теги, супроводжувати дані тегами
miscreant – злодій, негідник
port – переносити, адаптувати
spoof – підробляти, підміняти *(наприклад, адресу електронної пошти)*, імітувати
upload – пересилати *(інформацію в комп'ютер вищого рівня, наприклад, з локального комп'ютера – на сервер)*
account name – реєстраційне / облікове ім'я
DNS server – сервер служби імен доменів
credentials – мандат *(обліковий запис з параметрами доступу користувача, сформований після його успішної автентифікації)*
mastermind a ring – керувати злочинним угрупуванням
cybercriminal – кіберзлочинець, комп'ютерний злодій
craft – виготовляти, створювати
plant – 1) встановлювати, розміщувати; 2) ховати
hit (*p., pp.* hit) – завдавати шкоди, уражати
class-action law suit – колективний судовий позов
approximately – приблизно

Exercise 44. *Translate the word combinations below into Ukrainian.*

Data-stealing malware; to divest victims of personal and proprietary information; monetizing stolen data through direct use or underground distribution; content security threats; to include keyloggers, spyware, adware, backdoors, and bots; to refer to activities such as spam, phishing, DNS poisoning, SEO abuse; proxy information; to leave traces of the event; to be routinely flushed; a drive-by download process; to be generally temporary or rogue; to frequently change and extend the function; to use multiple file encryption levels; to thwart to record keystrokes, passwords, and screenshots; leakage protection to hinge on metadata tagging; to spoof pages of the bank website; to covertly monitor web-surfing habits; to steal login credentials; to mastermind a ring; to craft phishing emails; to plant additional malware

on users' PCs; a data security breach; to be hit by several class-action law suits.

Exercise 45. *Read and translate text 5.*

Text 5. Data-Stealing Malware

Data-stealing malware is a web threat that divests victims of personal and proprietary information with the intent of monetizing stolen data through direct use or underground distribution. Content security threats that fall under this umbrella* include keyloggers, spyware , adware, backdoors, and bots. The term does not refer to activities such as spam, phishing, DNS poisoning, SEO abuse, etc. However, when these threats result in file download or direct installation, as most hybrid attacks do, files that act as agents to proxy information will fall into the data-stealing malware category.

Characteristics of data-stealing malware:

Does not leave traces of the event.

- The malware is typically stored in a cache that is routinely flushed.
- The malware may be installed via a drive-by download process.
- The website hosting the malware as well as the malware is generally temporary or rogue.

Frequently changes and extends its function.

- It is difficult for antivirus software to detect final payload attributes due to the combinations of malware components.
- The malware uses multiple file encryption levels.

Thwarts Intrusion Detection Systems (IDS) after successful installation.

- There are no perceivable network anomalies.
- The malware hides in web traffic.
- The malware is stealthier in terms of traffic and resource use.

Thwarts disk encryption.

- Data is stolen during decryption and display.
- The malware can record keystrokes, passwords, and screenshots.

Thwarts Data Loss Prevention (DLP).

- Leakage protection hinges on metadata tagging, not everything is tagged.
- Miscreants can use encryption to port data.

* fall under this umbrella – належати до цієї категорії

Examples and incidents of data-stealing malware:

- Bancos, an info stealer that waits for the user to access banking websites then spoofs pages of the bank website to steal sensitive information.
- Gator, spyware that covertly monitors web-surfing habits, uploads data to a server for analysis then serves targeted pop-up ads.
- LegMir, spyware that steals personal information such as account names and passwords related to online games.
- Qhost, a Trojan that modifies the hosts file to point to a different DNS server when banking sites are accessed then opens a spoofed login page to steal login credentials for those financial institutions.
- Albert Gonzalez was accused of masterminding a ring to use malware to steal and sell more than 170 million credit card numbers in 2006 and 2007 – the largest computer fraud in history. Among the firms targeted were BJ's Wholesale Club, TJX, DSW Shoe, OfficeMax, Barnes & Noble, Boston Market, Sports Authority and Forever 21.
- A Trojan horse program stole more than 1.6 million records belonging to several hundred thousand people from Monster Worldwide Inc's job search service. The data was used by cybercriminals to craft phishing emails targeted at Monster.com users to plant additional malware on users' PCs.
- Customers of Hannaford Bros. Co, a supermarket chain based in Maine, were victims of a data security breach involving the potential compromise of 4.2 million debit and credit cards. The company was hit by several class-action law suits.
- The Torpig Trojan has compromised and stolen login credentials from approximately 250,000 online bank accounts as well as a similar number of credit and debit cards. Other information such as email, and FTP accounts from numerous websites, have also been compromised and stolen.

Exercise 46. *Find in text 5 the English for:*

шкідливі програми, що крадуть дані; позбавляти жертви особистої та службової інформації; перетворювати на гроші викрадені дані шляхом прямого використання або нелегального поширення; загрози безпеці контенту; включати логери клавіатури та шпигунське програмне забезпечення; безкоштовні програмні продукти, що містять рекламу, лазівки та мережеві агенти-роботи; стосуватися та-

ких процесів, як спам, фішинг та псування сервера імен доменів; неправомірне застосування оптимізатора пошукових систем; залишати ознаки подій; процес автоматичного завантаження непотрібної програми в комп'ютер; тимчасовий або некерований; використовувати кілька рівнів шифрування файлів; руйнувати систему виявлення мережевих атак; реєструвати хід клавіш, паролі та моментальні знімки екрану; перешкоджати попередженню втрати даних; залежати від тегування метаданих; підміняти сторінки вебсайтів з метою викрадення конфіденційної інформації; надсилати цільове спливаюче вікно; керувати злочинним угрупуванням; порушення системи захисту; колективний судовий позов.

Exercise 47. *Decipher the abbreviations below:*

DNS, SEO, IDS, DLP, FTP, malware, Inc., email, Co.

Exercise 48. *Answer the questions on text 5.*

1. What is the purpose of data-stealing malware? 2. What do content security threats include? 3. In what case do such activities as spam, phishing, DNS poisoning, SEO abuse fall into the data-stealing malware category? 4. What are the characteristics of data-stealing malware? 5. What are the examples and incidents of data-stealing malware?

Exercise 49. *Speak on data-stealing malware. Give additional information about examples and incidents of this malware you have read or heard of.*

Exercise 50. *Define the parts of speech that the words below belong to. Name noun-, adjective-, adverb- and verbal-forming suffixes.*

Security, confidentiality, integrity, availability, identity, anonymity, property, activity, application, implication, modification, authentication, nonrepudiation, remediation, technology, strategy, methodology, privacy, programmer, operator, server, intruder, development, trustworthiness, service, disclosure, meaning, access, use, control, motivate, communicate, protect, preventing, detecting, accessing, physical, national, technical, conceptual, isolated, crippled, unauthorized, multifaceted, elusive, objective, sensitive, collective, significant, important, electronic, easy, relatively, easily, simply, constantly, respectively, deliberately, conversely.

Exercise 51. *Read and translate text 6.*

Text 6. Safe Computing Rules

1. Ensure that any message sent arrives at the proper destination.
2. Ensure that any message received was in fact the one that was sent, (nothing added or deleted).
3. Control access to your network and all its related parts, (this means terminals, switches, modems, gateways, bridges, routers, and even printers).
4. Protect information in-transit, from being seen, altered, or removed by an unauthorized person or device.
5. Any breaches of security that occur on the network should be revealed, reported and receive the appropriate response.
6. Have a recovery plan, should both your primary and backup communications avenues fail.
7. Use and update antivirus software regularly.
8. Scan any newly received disks and files before loading, opening, copying, etc.
9. Never assume disks and/or files are virus-free.
10. To help avoid boot viruses, do not leave diskettes in your computer when shutting it down.
11. Change your computer's SMOS boot sequence to start with the C drive first, then the A drive.

For offices or homes with one or two computers, following these basic rules faithfully is probably adequate protection. However, in organizations with multiple PCs, especially in networks, a sound antivirus strategy will necessarily be more complex. This is because vulnerability to viruses increases in proportion to the number of machines, the extent of their interconnection, and the number of non-technical users who may view antivirus vigilance as "someone else's job". (In contrast, a solo entrepreneur is likely to take the virus threat seriously because he or she will have to deal with infection results personally or pay an outside consultant.) All organizations are different in the way they operate and the industries they serve, so no one antivirus scheme is correct for all enterprises. However, at the very least, a company's program should include ongoing user education and a system for tracking virus activity (suspect and real) in addition to using antivirus software. Ultimately, your goal is to provide consistent, effective protection and a "damage

control and recovery" plan for virus infections that may occur despite your efforts. In addition, and perhaps most importantly, you want to achieve this while minimizing any negative impact on staff productivity and system/network resources. Therefore, to formulate a comprehensive antivirus plan, it is necessary to first analyze the "bit picture" of your organization along with its more detailed computing characteristics.

Exercise 52. *Find in text 6 the English for:*

контролювати доступ до вашої мережі та пов'язаних з нею частин (терміналів, перемикачів, модемів, шлюзів, мостів, маршрутизаторів і навіть принтерів); будь-які порушення безпеки, що трапляються в мережі; план відновлення; регулярно використовувати та оновлювати антивірусне програмне забезпечення; уникати вірусів під час завантаження; надійна антивірусна стратегія; вразливість до вірусів; пропорційно кількості машин; антивірусна пильність; серйозно сприймати загрозу інфікування вірусом; постійне навчання користувачів; забезпечувати стійкий, ефективний захист; план контролю уражень та відновлення; зменшити негативний антивірусний вплив на продуктивність персоналу та ресурсів системи (мережі); всеохопний план антивірусного захисту.

Exercise 53. *Choose between the word-combinations* ***as well*** *and* ***as well as*** *in the sentences below.*

1. At the conference there were other software engineers … . 2. Professor Sidorov gave us lectures in programming languages, software engineering … in computer architecture. 3. Alan Turing … Ada Lovelace, George Boole, and Charles Babbage has contributed greatly to the development of computer science. 4. Along with the advantages of the new system were discussed its possible weak points … . 5. In machine code instructions … data are represented as sequences of zeros and ones. 6. The hard coating technology used for BD ROM discs can be applied to rewritable and recordable Blu-ray Discs … .

Exercise 54. *Choose the correct form of the verbs in brackets and translate the sentences.*

1. People (to worry) about the security of their computers for many years.

a) have worried
b) worried

c) has worried
d) is worried

2. Confidentiality means that information cannot (to access) by unauthorized parties.

a) accessed
b) was accessed
c) be accessed
d) will access

3. Prevention measures help to stop unauthorized users from (to access) any part of your computer system.

a) access
b) accessing
c) being accessed
d) having accessed

4. The need for keeping information secret (to arise) from the use of computers in sensitive fields.

a) arises
b) is arisen
c) arise
d) is arising

5. The major technical areas of computer security are usually (to represent) by the initials CIA.

a) represent
b) has been represented
c) is represented
d) are represented

6. Military and civilian institutions in the government often (to restrict) access to information to those who (need) that information.

a) restricted, will need
b) restrict, need
c) have restricted, needed
d) is restricted, needed

Exercise 55. *Translate into English paying special attention to the italicized words.*

1. ***Захист інформації*** передбачає застосування ***запобіжних заходів***, що включають ***сукупність процесів і механізмів***, які, з одного боку, ***забороняють неправочинним користувачам доступ до інформації***, а з іншого – ***залишають її доступною і корисною*** для ***обраних користувачів***. 2. ***Питання комп'ютерної безпеки*** нині є важливим чинником ***розвитку та застосування комп'ютерних технологій у всьому суспільстві,*** оскільки зараз ***захистити інформацію і власність від викрадення або пошкодження*** неможливо, ***просто обмеживши фізичний доступ,*** як це було ***на ранній стадії розвитку комп'ютерів***. 3. ***Порушення цілісності баз даних*** може статися внаслідок як ***незаконної діяльності,*** так і втручання ***ненадійних осіб,*** що може призвести до ***небажаної поведінки комп'ютера*** та пов'язаних з цим ***незапланованих подій.*** 4. ***Забезпечення контролю доступу*** дозволяє ***визначити, чи намагався хтось «зламати» вашу систему*** з метою ***незаконного використання***. 5. Якщо ***на ранній стадії свого розвитку*** комп'ютери ***мали здебільшого військове застосування,*** нині багато як ***військових, так і цивільних установ*** зберігають у своїх комп'ютерах ***конфіденційну та цінну інформацію***. 6. Терміном «цілісність інформації» зазвичай називають ***достовірність даних або ресурсів*** і ***запобігання їх недоречним або несанкціонованим змінам***. 7. Можна стверджувати, що лише ***непідімкнений до мережі комп'ютер*** є цілковито захищеним від втручання ***неправочинних користувачів,*** та й то тільки тоді, якщо фізичний доступ до нього обмежений ***обраними користувачами***. 8. У низці ***пов'язаних із секретністю галузей,*** таких як: оборона, дипломатія, окремі наукові дослідження, ***доступ до інформації обмежується*** з міркувань державної безпеки. 9. У наш час комп'ютери широко застосовують у всіх галузях людської діяльності, і зараз неможливо ***точно підрахувати*** ні кількість комп'ютерів на планеті, ні кількість ***щоденних користувачів Інтернету***. 10. Зараз, в добу Інтернету, науковець у своїй дослідницькій діяльності може збирати ***відповідну інформацію*** швидко та легко ***з різних джерел***, які донедавна було важко уявити. 11. ***Неправочинні суб'єкти***, тобто особи, які незаконно проникли у вашу систему, можуть ***зумисно влаштувати заборону доступу до даних***, зробивши ***систему недоступною***. 12. Комп'ютерна злочинність, яка нині набула значних масштабів, змушує вживати ***запобіжних заходів*** навіть до тих користувачів, які не (три)мають у своїх комп'ютерах ***конфіденційної або цінної інформації***.

Exercise 56. *Do the following assignments.*

1. Add some more safe computing rules to those listed in text 6.
2. What antivirus software do you prefer? Why? Share your experience in using antivirus software with your group-mates.
3. Discuss the advantages and disadvantages of different antivirus software.

UNIT 12. SOFTWARE AND DATA SECURITY

Exercise 1. *Study the basic vocabulary.*

a) terms
software security – захист / безпека програмного забезпечення
buffer overflow – переповнення буферу
protection failure – відмова захисту
design compartmentalization – роздробленість проектування
fragility – вразливість
circumvention – обхід
script – сценарій, скрипт
buggy code – код, що містить помилки
abuse case – випадок неправильного / зловмисного використання
scenario – сценарій, план дій
attack pattern – схема атаки
use case – варіант використання
architectural risk analysis – аналіз архітектурного ризику
penetration testing – випробування на можливість проникнення до системи, тест на захист від несанкціонованого доступу
risk-based security testing – тестування захищеності, що базується на оцінюванні ризиків
operational security – безпека в експлуатації

b) nouns
misuse – неправильне застосування / використання
practices – інструкції
touch point of sth – *тут:* точка дотику (*із чимось*); питання, що стосується чогось
ambiguity – неоднозначність
mitigation – пом'якшення *(згубних наслідків)*

severity – серйозність, критичність
precaution – запобіжний захід

c) verbs
reinforce, enforce – підсилювати, посилювати, зміцнювати, збільшувати
go about – займатися *(чимось)*, братися / взятися *(до чогось)*
make sure – переконатися, пересвідчитися

d) adjectives
complicated – складний
weak – слабкий
well-documented – переконливо підтверджений документальними доказами
tangible – відчутний

Exercise 2. *Choose nouns among the following words. Put the first letters of the nouns into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Possible, continuous, penetration, complicated, requirement, untrustworthy, avoid, optimization, clarify, template, eliminate, obvious, error, detect, crime, reinforce, weak, tradition, architectural, tangible, buggy, implementation, occur, properly, observation, apply, incorporate, network.

Exercise 3. *Give synonyms (a) and antonyms (b) for the following words:*

a) means, engineering, continue, complicated, bug, fragility, properly, improve, secure, widely, developer, attacker, critical, incorporate, clearly, define, clarify, detect, eliminate, obvious, trustworthy, perform, consider, prioritize, security break, apply, provide, tangible, contain;

b) trustworthy, input, simple, tangible, secure, correctly, easy, encrypt, possible, external, weak, obvious, include.

Exercise 4. *Write derivatives of the words below and explain their meanings.*

Model: consider – consideration – considerate – considerable – considerably

Consider, correct, vulnerable, trust, architect, fragile, identify, fail, bug, clear, compartment, require, behave, penetrate, weak, differ, implement, continue, operate, describe, provide, secure, force.

Exercise 5. *Give Ukrainian equivalents for the following word combinations.*

Software security; the means of engineering software; to continue functioning correctly under malicious attack; to be reinforced by the continuous growth of software vulnerabilities; to double each year; complicated problems; implementation bugs such as buffer overflows, race conditions, and untrustworthy input problems; architectural flaws such as protection failures, misuse of cryptography and broken access control; along with design compartmentalization and fragility; tools to identify bugs; to enforce secure coding; to contain well-documented design flaws; to allow easy circumvention of encryption; to consider available examples of configurations, scripts, and buggy code to avoid repeating mistakes; in order to understand how to build secure software; breaking code; to be incorporated at critical points within the development life cycle; the touch points of software security; critical data elements and user authentication; to define abuse cases to describe scenarios; important non-normative behaviour that should not be allowed to occur; possible attack patterns, requirements, and use cases; to clarify what should not be allowed to happen; to review code; to detect security vulnerabilities; to eliminate obvious errors; to perform an architectural risk analysis; to consider resistance to attack, design ambiguity, and design weakness (fragility); to prioritize identified risks for mitigation; penetration testing; to evaluate the operational severity of risks; to perform risk-based security testing; to concentrate on ensuring that security breaks won't occur; to make sure obvious vulnerabilities are not missed; applying a few relatively simple precautions; to make a tangible difference.

Exercise 6. *Read and translate text 1.*

Text 1. Software Security Problems and Practices

Software security is the means of engineering software so that it continues functioning correctly under malicious attack. The security problem is reinforced by the continuous growth of software vulnerabilities that more than double each year. Software security problems are complicated. At a high level there are two major categories: (1) implementation bugs such as buffer overflows and untrustworthy input problems and (2) architectural flaws such as protection failures, misuse of cryptography, broken access control, along with design compartmentalization and fragility. Software security is about building things properly.

Tools to identify bugs are improving, but popular languages such as C and C++ do not enforce secure coding. Widely used communication protocols are weak; one example is the 802.11b WEP protocol which contains well-documented design flaws that allow easy circumvention of encryption. Developers must learn from past mistakes so as to better write secure code. They must consider available examples of configuration, scripts, design flaws, and buggy code to avoid repeating mistakes.

In order to understand how to build secure software, the developer must understand how an attacker goes about breaking code. These are the best practices that should be incorporated at critical points within the development life cycle (the touch points of software security):

• Clearly describe the security needs of functional requirements such as protection requirements for critical data elements and user authentication.

• Define abuse cases to describe scenarios of important non-normative behaviour that should not be allowed to occur. Build an attack model from possible attack patterns, requirements, and use cases to clarify what should not be allowed to happen (anti-requirements).

• Review code with a tool that detects security vulnerabilities to eliminate obvious errors.

• Perform an architectural risk analysis to consider resistance to attack, design ambiguity, and design weakness (fragility). Prioritize identified risks for mitigation.

• Use penetration testing to evaluate the operational severity of risks identified in risk analyses.

• Perform risk-based security testing using the abuse cases and architectural risk review to identify critical test scenarios. Concentrate on ensuring that security breaks won't occur.

• Involve operational security resources to apply their knowledge and provide for external review to make sure obvious vulnerabilities are not missed.

These touch points should be introduced into any software development life cycle. Even applying a few relatively simple precautions can make a tangible difference in moving along the path toward secure software. Now everybody agrees that software security is central to computer security and it is time to put this philosophy into practice.

Exercise 7. *Find in text 1 the English for:*

захист програмного забезпечення; засіб розроблення програмного забезпечення; продовжувати правильно функціонувати за умови

зловмисної дії; постійне збільшення слабких місць у програмному забезпеченні; зростати більш ніж удвічі щороку; помилки реалізації, переповнення буферу; проблеми, зумовлені ненадійністю вхідної інформації; архітектурні дефекти; відмова захисту; неправильне використання криптографії; порушений контроль доступу; роздробленість проектування та вразливість; підвищувати надійність кодування; уникати повторення помилок; зламати код; питання, що стосуються захисту програмного забезпечення; важливі елементи даних; визначати випадки неправильного використання; з'ясувати, чого не можна допускати; перевіряти код; виявляти слабкі місця захисту; усунути очевидні помилки; здійснити аналіз архітектурних ризиків; враховувати протидію атакам та неоднозначність конструкції; упорядковувати виявлені ризики відповідно до пріоритетів з метою запобігання негативних наслідків; оцінювати функціональну серйозність виявлених ризиків; визначати важливі сценарії тестування; зосередитися на методах запобігання порушенням безпеки; відносно прості запобіжні заходи; відчутна різниця.

Exercise 8. *Say whether the statements below are true or false. Correct the false ones.*

1. Software security is the means of modelling software so that it continues functioning correctly under malicious attack. 2. The security problem is reinforced by the continuous growth of software vulnerabilities that more than double each 5 years. 3. Software security problems are complicated. 4. At a low level there are two major categories: (1) implementation bugs such as buffer overflows and untrustworthy input problems and (2) architectural flaws such as protection failures, misuse of cryptography, broken access control, along with design compartmentalization and fragility. 5. Tools to identify bugs are improving and popular languages such as C and C++ enforce secure coding. 6. Developers must learn from past mistakes so as to better write secure code, they must consider available examples of configuration, scripts, design flaws, and buggy code to avoid repeating mistakes. 7. In order to understand how to build secure software, the developer must understand how an attacker goes about breaking code. 8. Some touch points of software security include: describing the security needs of functional requirements, defining abuse cases to describe scenarios of important non-normative behaviour and reviewing a code with a tool that detects secu-

rity vulnerabilities. 9. Applying a few relatively simple precautions cannot make a tangible difference in moving along the path toward secure software. 10. Now everybody agrees that software security is central to computer security and it is time to put this philosophy into practice.

Exercise 9. *Form all possible word combinations with the words from both columns. Translate them.*

1) to understand	a) abuse cases
2) to be incorporated at	b) secure coding
3) to enforce	c) relatively simple precautions
4) to define	d) critical points
5) to allow	e) how to build secure software
6) to detect	f) obvious errors
7) to learn from	g) easy circumvention of encryption
8) to eliminate	h) resistance to attack
9) to consider	i) past mistakes
10) to apply	j) security vulnerabilities

Exercise 10. *Fill in the blanks with prepositions* ***from, with, by, under, of, at, in, about, within, into, along, to, toward*** *where necessary.*

1. Software security is the means … engineering software so that it continues functioning correctly … malicious attack. 2. The security problem is reinforced … the continuous growth … software vulnerabilities that more than double each year. 3. … a high level there are two major categories. 4. Software security is connected … building things properly. 5. Developers must learn … past mistakes so as to better write secure code. 6. … order to understand how to build secure software, the developer must understand how an attacker goes … breaking code. 7. There are the best practices that should be incorporated … critical points … the development life cycle (the touch points … software security). 8. Clearly describe the security needs … functional requirements … critical data elements and user authentication. 9. These touch points should be introduced … any software development life cycle. 10. Applying a few relatively simple precautions can make a tangible difference … moving … the path … secure software. 11. Software security is central … computer security and it should be put … practice.

Exercise 11. *Fill in the blanks with proper terms (**data security, penetration testing, software development life cycle, software security, implementation bug, architectural flaw, software bug, computer security, vulnerability**) to complete the sentences.*

1. _____is a bug such as buffer overflows and untrustworthy input problems. 2. _____ is a branch of computer technology that deals with preventing and detecting unauthorized use of your computer. 3. _____ is the inability to withstand the effects of a hostile environment. 4. _____ is a protection failure, misuse of cryptography, broken access control, design compartmentalization and fragility. 5. _____is the means of ensuring that data is kept safe from corruption and that access to it is suitably controlled. 6. _____ is an error, flaw, failure, or fault in a computer program or system that produces an incorrect or unexpected result, or causes it to behave in unintended ways. 7. _____ is a sequence of steps aimed at creating software including specifying requirements, design, coding and verification. 8. _____ is the means of engineering software so that it continues functioning correctly under malicious attack. 9. _____is a method of evaluating computer and network security by simulating an attack on a computer system or network from external and internal threats.

Exercise 12. *Answer the questions on text 1.*

1. What is software security? 2. What is the software security problem reinforced by? 3. What are the high level software security problems? 4. What is the role of popular languages such as C and C++ in enforcing secure coding? 5. What must developers learn from so as to better write secure code and what must they consider? 6. What must a developer understand in order to build secure software? 7. What should be incorporated at critical points within the development life cycle? 8. What security needs of functional requirements should be clearly defined? 9. What is an attack model built from? 10. What is code reviewed for? 11. What is the purpose of architectural risk analysis? 12. What is penetration testing used for? 13. What must be performed to identify critical test scenarios? 14. What must operational security resources be involved for? 15. What can make a tangible difference in moving along the path toward secure software? 16. Do you agree that software security is central to computer security?

Exercise 13. *Put all possible questions to the sentences below.*

1. The security problem is reinforced by the continuous growth of software vulnerabilities. 2. At a high level there are two major categories of software security. 3. Popular languages such as C and C++ do not enforce secure coding. 4. Developers must learn from past mistakes so as to better write secure code. 5. The best practices should be incorporated at critical points within the development life cycle. 6. Now everybody agrees that software security is central to computer security.

Exercise 14. *Translate into English.*

1. Захист програмного забезпечення – це засіб розроблення програмного забезпечення, який гарантує, що воно продовжуватиме правильно функціонувати у разі зловмисних дій. 2. Проблема безпеки посилюється через постійне збільшення вразливих місць у програмному забезпеченні. 3. Помилки реалізації включають переповнення буферу та проблеми, зумовлені ненадійністю вхідної інформації. 4. Архітектурні помилки включають помилки захисту, неправильне застосування криптографії, порушення контролю доступу разом з роздробленістю суб'єктів проектування та вразливістю. 5. Засоби виявлення помилок вдосконалюються, але такі популярні мови, як C та C++ не підвищують надійності кодування. 6. Поширені комунікаційні протоколи є слабкими, вони містять дефекти проектування, переконливо підтверджені документально, але дозволяють легко провести розшифрування. 7. Розробники повинні враховувати наявні зразки конфігурації, сценарії, дефекти проектування та помилкові коди, щоб уникнути повторення помилок. 8. Щоб побудувати надійне програмне забезпечення, розробник має зрозуміти, яким чином зловмисник зламує код. 9. Ось найкращі методи, які треба ввести до ключових точок у процесі життєвого циклу розроблення (точок дотику, що стосуються захисту програмного забезпечення). 10. Потрібно чітко описати потреби функціональних вимог безпеки, таких, як вимоги захисту важливих елементів даних та автентифікації користувача. 11. Необхідно визначити випадки неправильного використання, щоб описати сценарії нештатної поведінки, яким треба запобігти. 12. Перевірте код інструментом, що виявить слабкі місця безпеки для того, щоб усунути очевидні помилки. 13. Зробіть аналіз архітектурного ризику, щоби врахувати протидію атаці, неоднозначність та слабкість конструкції. 14. Упорядкуйте виявлені ризики відповідно до пріоритетів з ме-

тою запобігання негативних наслідків. 15. Застосовуйте тест на захист від несанкціонованого доступу, щоб оцінити функціональну серйозність виявлених ризиків. 16. Виконайте тестування захищеності, яке базується на оцінюванні ризиків, використовуючи випадки неправильного застосування та перевірку архітектурних ризиків, щоб визначити важливі сценарії тестування. 17. Такі точки дотику мають бути введені до будь-якого життєвого циклу розроблення програмного забезпечення. 18. Застосування навіть кількох відносно простих запобіжних заходів може відчутно поліпшити захист програмного забезпечення. 19. Зараз всі погоджуються з тим, що захист програмного забезпечення є головним у комп'ютерній безпеці, і настав час реалізувати цю філософію практично.

Exercise 15. *Retell the text "Software Security".*

Exercise 16. *Read and translate text 2.*

Text 2. Building Secure Software

External faults that threaten the software's dependable operation are seen as a security issue when (1) the faults result from malicious intent or (2) the faults, regardless of their cause, make the software vulnerable to threats to its security. Security is about preventing adverse consequences from the intentional and unwarranted actions of others.

To be considered secure, software must exhibit three properties:

1. **Dependability:** Dependable software executes predictably and operates correctly under all conditions, including hostile conditions, including when the software comes under attack or runs on a malicious host.

2. **Trustworthiness:** Trustworthy software contains few if any vulnerabilities or weaknesses that can be intentionally exploited to subvert or sabotage the software's dependability. In addition, to be considered trustworthy, the software must contain no malicious logic that causes it to behave in a malicious manner.

3. **Survivability (also referred to as "Resilience"):** Survivable or resilient software is software that is resilient enough to (1) either resist (i.e., protect itself against) or tolerate (i.e., continue operating dependably in spite of) most known attacks plus as many novel attacks as possible, and (2) recover as quickly as possible, and with as little damage as possible, from those attacks that it can neither resist nor tolerate.

The objective of secure software development is to design, implement, configure, and sustain software systems in which security is a necessary property from the beginning of the system's life cycle (i.e., needs and requirements definition) to its end (retirement). Experience has taught that the most effective way to achieve secure software is for its development life cycle processes to rigorously conform to secure development, deployment, and maintenance principles and practices. Organizations that have adopted a secure software development life cycle (SDLC) process have found almost immediately upon doing so that they have begun finding many more vulnerabilities and weaknesses in their software early enough in the SDLC that they are able to eradicate those problems at an acceptable cost. Moreover, as such secure practices become second nature over time, these same developers start to notice that they seldom introduce such vulnerabilities and weaknesses into their software in the first place.

Exercise 17. *Answer the questions on text 2.*

1. When are external faults that threaten the software's dependable operation seen as a security issue? 2. What properties must software exhibit to be considered secure? 3. What is dependable software? 4. What must software contain to be considered trustworthy? 5. What is survivable or resilient software? 6. What is the objective of secure software development? 7. What is the most effective way to achieve secure software?

Exercise 18. *Write out all Non-Finite forms of the verb (Infinitives, Participles and Gerunds) from the text of Exercise 16. Identify their forms and functions.*

Exercise 19. *Translate the following sentences. Define the forms and functions of all Non-Finite forms of the verb.*

1. Many of Microsoft software products are predominantly designed using the "trusted system" approach. 2. Malicious software is software designed to infiltrate a computer system without the owner's consent. 3. Having attached itself to executable files or documents, the virus propagates to other programs on the computer. 4. Spyware programs installed as Trojan horses are commercially produced for gathering information about computer users. 5. Having been attacked by hackers, the program lost some encrypted data. 6. Science fiction is fiction dealing with imagined scientific discoveries and advances. 7. The main rea-

son for adding security practices throughout the Synchronous Data Link Control (SDLC) is to establish a software life cycle process codifying both caution and intention. 8. Creating a secure development community using collaboration technologies and a well-integrated development environment promotes a continuous process of improving and focusing on secure development life cycle principles and practices. 9. Not having found necessary data they used new information technologies. 10. The presence of automated assistance to the controllers should reduce the need for coordination. 11. A programming language is a special coding system designed for writing instructions for a computer. 12. Having been invented for solving mathematical problems, FORTRAN allowed programmers to write algebraic expressions.

Exercise 20. *Translate the sentences paying attention to the Infinitive, Participial and Gerundial Complexes.*

1. Everybody agrees with software security being central for computer security. 2. They saw the delegation enter the University. 3. A firewall being installed, spyware cannot gain access to the internal network. 4. We know of Windows having been developed by Microsoft. 5. He is said to become a good programmer. 6. The specialist was considered being experienced in the area of human-computer interaction. 7. Their following this principle minimized software vulnerability to attack. 8. The teacher wanted him to write a new computer program. 9. The programmer was seen reviewing and debugging the code. 10. We rely on this device operating properly. 11. A machine language is known to be a low-level language. 12. Computer scientists were considered developing new high-level languages. 13. I insist on this software being debugged immediately. 14. Data processing was considered finished. 15. I want you to be more attentive at the lectures.

Exercise 21. *Memorize the following types of conditional sentences.*

I. Real situation in the future	1) If I **am** not busy, I **will help** you with the problem. 2) If we **debug** the program, it **will work** correctly	1) Якщо я не буду зайнятий, то допоможу тобі із задачею. 2) Якщо ми налагодимо програму, то вона буде працювати правильно

II. Unreal situation in the present/ future	1) If I **were (was)** not busy tomorrow, I **would help** you with the problem. 2) If we **debugged** the program, it **would work** correctly	1) Якби я не був завтра зайнятий, то допоміг би тобі з задачею. 2) Якби ми налагодили програму (сьогодні), то вона б працювала правильно
III. Unreal situation in the past.	1) If I **had not been** busy yesterday, I **would have helped** you with the problem. 2) If you **had worked** systematically, you **would not have failed** the exam	1) Якби я не був зайнятий вчора, то допоміг би тобі з задачею. 2) Якби ти систематично працював, то не провалив би екзамен
IV. Mixed type (combination of unreal situation in the present and in the past)	1) If I **were (was)** as clever as you, I **would not have failed** the exam. 2) If we **had debugged** the program yesterday, it **would work** correctly (today)	1) Якби я був такий розумний, як ти, то не провалив би екзамен. 2) Якби ми налагодили програму вчора, то вона б працювала правильно (сьогодні)

Exercise 22. *Complete the sentences making use of all possible types of conditionals. Translate them.*

1. If you (know) English well, you (work) abroad. 2. If you (study) well, you (be) a good specialist. 3. If you (be) an experienced worker, you (can make) a good career. 4. If you (have) a good memory, you (can learn) easily many programming languages. 5. If you (know) a lot of programming languages, you (find) a good job. 6. If you (install) a virus scanner, your computer (be) protected. 7. If antivirus software periodically (receive) virus definition, updates of the software effectiveness (be maintained). 8. If the resulting software (comprise) well defined, independent components, that (lead) to better maintainability.

Exercise 23. *Complete the sentences using the verbs in the proper form. Translate the sentences.*

1. If the problem (be) not well-defined, it (result) in poor design. 2. If you (work) harder, you (make) more progress. 3. If he (work) harder yesterday,

he (make) fewer mistakes. 4. If I (have) a dictionary, I (translate) the article yesterday. 5. If you (not work) systematically, you (fail) the exam. 6. The teacher (be) displeased if you (not do) the laboratory work. 7. If you (be) a professional associated with design and drafting, you (find) this book informative. 8. If you (widen) your knowledge about CAD applications, you (change) drawings with minimal effort. 9. The experiment (give) good results if you (have) more accurate input. 10. If the program (be) maintained, it (work) properly. 11. If you (write) the program yesterday, we (begin) debugging and testing just now. 12. If instructions (be) translated into machine language, they (be) processed by the computer. 13. If your program (be) very small, you (skip) the step of designing. 14. If you (be) more experienced, you (find) this step very useful.

Exercise 24. *Analyse the following examples of subjunctive mood usage.*

1. I wish I **were (was)** more successful at the exam	1. Шкода, що мені не дуже пощастило на екзамені. (Добре було б, якби мені більше пощастило на екзамені)
2. I wish I **lived** in the hostel	2. Шкода, що я не живу в гуртожитку. (Добре було б, якби я жив у гуртожитку)
3. I wish you **had translated** the article yesterday	3. Шкода, що ви не переклали статтю вчора. (Добре було б, якби ви переклали статтю вчора)
4. I wish she **would come** to the university tomorrow	4. Шкода, що вона не прийде в університет завтра. (Добре було б, якби вона прийшла в університет завтра)
5. I wish you **could give** the verified data to me next week	5. Шкода, що ти не даси мені перевірені дані наступного тижня. (Добре було б, якби ти дав мені перевірені дані наступного тижня)

Exercise 25. *Complete the sentences using the verbs in brackets in the proper form. Translate the sentences.*

1. I wish I (can) speak English better. 2. I wish I (pass) my module test last month. 3. He wishes he (not break) the system. 4. I wish I (be) a software developer. 5. I wish you (read) more in future. 6. I wish I (know) Spanish. 7. She wishes she (learn) the lesson. 8. I wish I never (suggest) this idea. 9. He wishes he (earn) more money. 10. I wish he

(write) the composition yesterday. 11. I wish it (be) not late to go to the conference. 12. I wish you (know) physics better. 13. My friend wishes he (enter) the University. 14. I wish you (have) more practice in developing software. 15. I wish we (go) to the library yesterday. 16. We wish we (buy) a book on software design. 17. He wishes he (join) the English speaking club last month. 18. I wish you (make) fewer mistakes in the dictation. 19. I wish you (start) the experiment already. 20. I wish this notebook (be) not so expensive.

Exercise 26. *Translate into English.*

1. Було б добре, якби зараз були канікули. 2. Якби я був на вашому місці, то порадився б з батьками. 3. Якби ви допомогли мені розв'язати цю задачу, то я був би вам дуже вдячний. 4. Шкода, що ви запізнилися на лекцію. 5. Якби ми отримали інструкції вчора, то зробили б роботу вчасно. 6. Якби я знав французьку мову, то уже давно б поговорив з нею. 7. Якби у нас було більше практичних занять з англійської мови, ми б краще знали цей предмет. 8. Шкода, що нам раніше не спало на думку прошукати книгу в бібліотеці. 9. Якби я був успішним програмістом, то працював би за кордоном. 10. Шкода, що вона не прийде завтра на заняття. 11. Якби цей метод проектування був більш детально вивчений, його б широко застосовували практично.

Exercise 27. *Play a game. Look at this sequence of sentences, then use the prompts below to act out similar hypotheses.*

S1: If computers were more powerful, they would replace people.
S2: If people were replaced by computers, they would have more free time.
S3: If they had more free time, they would go on holiday.
S4: If they went on holiday…

- If I won $ 1,000,000 …
- If I were an experienced software engineer …
- If I studied abroad…

Exercise 28. *Study the vocabulary to text 3.*

Data erasure – стирання даних, руйнування інформації
disk encryption software – програмні засоби шифрування диску
disk encryption hardware – апаратні засоби шифрування диску

on-the-fly encryption – оперативне шифрування, шифрування в реальному часі

transparent encryption – «прозоре» (непомітне) шифрування *(не залежить від характеристик системи і не впливає на її нормальне функціонування)*

hardware-based security – апаратний захист; захист, що забезпечують апаратні засоби

security solution – рішення про забезпечення захисту

tamper – 1) зламувати 2) псувати 3) фальшувати, підробляти

security token – маркер доступу, токен *(фізичний пристрій, який надається авторизованому користувачеві з метою сприяння автентифікації)*

personal identification number (PIN) – особистий ідентифікаційний номер, ПІН-код

dongle – 1) (*електронний*) захисний ключ-заглушка *(під'єднується до порту введення–виведення)*; 2) електронний пристрій захисту *(вбудований у комп'ютер)*

log in – реєструватися в системі / мережі

log out – виходити із системи / мережі

privilege level – рівень привілеїв, рівень доступу

privileged operation – привілейована операція

unauthorized personnel – сторонній персонал, сторонні особи

overwriting – перезапис

steal (*p.* stole, *pp.* stolen) – красти

obscure – приховувати, робити непомітним

expose – 1) робити видимим, показувати; виставляти, показувати 2) піддавати дії

reside – перебувати, міститися

controller – контролер, пристрій керування

policies – заходи

representative – представник

software release – версія програмного забезпечення

vendor – постачальник, продавець

overwriting – перезапис

asset – ресурс, цифровий об'єкт

retire – *тут:* видаляти

proof – непроникний, непробивний, міцний

hence – тому, звідси

therefore – тому, отже

Exercise 29. *Choose verbs among the following words. Put the first letters of the verbs into the cells in the same order. Read and translate the word. Try to compose a similar exercise yourself.*

Outsourcing, ensure, suitably, unauthorized, notify, vulnerable, solution, compromise, transparent, asset, retire, proof, however, yield, privileged, protect, malicious, similar, tamper, privacy, identify, purpose, erasure, obtain, masking, token, negotiate.

Exercise 30. *Give synonyms (a) and antonyms (b) for the following words:*

a) privacy, security, erase, malicious, login, vendor, customer, personnel, asset, ensure, corrupt, require, gain, enable, reside, obscure, purpose, therefore, suitably, safe, means, refer to as, hacker, completely, compromise, resilience, resist, withstand, developer, mask, tamper;

b) authorized, usable, privacy, safe, login, personal, recoverable, secure, manual, possible, sensitive, transparent, malicious, weakness, effective, necessary, few, cause.

Exercise 31. *Write derivatives of the words below and explain their meanings.*

Model: intent – intention – intentional – intentionally

Intent, use, safe, suit, similar, author, source, represent, recover, solve, encrypt, corrupt, differ, access, combine, expose, inform, maintain, produce, complete, survive, object, secure, weak, threat, depend, predict, execute, develop, use.

Exercise 32. *Translate the word combinations below into Ukrainian.*

To be suitably controlled; to ensure privacy; backups, on-the-fly encryption or transparent encryption; to prevent data from being stolen; a malicious program; to corrupt the data in order to make it unrecoverable or unusable; security tokens; in order to be compromised; to gain physical access; to allow a user to login, logout and to set different privilege levels by doing manual actions; the current state of a user; to be vulnerable to malicious attacks by viruses and hackers; after a malicious access is obtained; to perform unauthorized privileged operation; to protect the operating system image and file system

privileges from being tampered; to ensure recovery of the data which has been lost; the process of obscuring (masking) specific data within a database table or cell; to ensure that data security is maintained and sensitive information is not exposed to unauthorized personnel; outsourcing vendors; data erasure; data residing on a hard drive or other digital media; to ensure that no sensitive data is leaked when an asset is retired or reused.

Exercise 33. *Read and translate text 3.*

Text 3. Data Security

Data security is the means of ensuring that data is kept safe from corruption and that access to it is suitably controlled. Thus data security helps to ensure privacy. It also helps in protecting personal data. The technologies of data security are: disk encryption, hardware-based mechanisms for protecting data, backups, data masking and data erasure.

Disk encryption is the technology that encrypts data on a hard disk drive. Disk encryption software or disk encryption hardware can be used for this purpose. This technology is often referred to as on-the-fly encryption or transparent encryption.

Software-based security solutions encrypt the data to prevent data from being stolen. However, a malicious program or a hacker may corrupt the data in order to make it unrecoverable or unusable. Similarly, encrypted operating systems can be corrupted by a malicious program or a hacker, making the system unusable. Hardware-based security solutions can prevent access to data and hence offer very strong protection against tampering and unauthorized access.

Hardware-based computer security offers an alternative to software-only computer security. Security tokens may be more secure due to the physical access required in order to be compromised. Access is enabled only when the token is connected and correct PIN is entered. However, dongles can be used by anyone who can gain physical access to the system. Newer technologies in hardware-based security solve this problem offering proof security for data.

The hardware-based security uses a device that allows a user to log in, log out and to set different privilege levels by doing manual actions. The device uses biometric technology to prevent malicious users from logging in, logging out, and changing privilege levels. The current state

of a user of the device is read by controllers in peripheral devices such as hard disks. Illegal access by a malicious user or a malicious program is interrupted based on the current state of a user by hard disk and DVD controllers. Hardware-based access control is more secure than protection provided by the operating systems as operating systems are vulnerable to malicious attacks by viruses and hackers. The data on hard disks can be corrupted after a malicious access is obtained. With hardware based protection, it is impossible for a hacker or a malicious program to gain access to secure data or perform unauthorized privileged operations. The hardware protects the operating system image and file system privileges from being tampered. Therefore, a completely secure system can be created using a combination of hardware-based security and secure system administration policies.

Backups are used to ensure recovery of the data which has been lost.

Data masking is the process of obscuring (masking) specific data within a database table or cell to ensure that data security is maintained and sensitive information is not exposed to unauthorized personnel. This may include masking the data from users (for example, banking customer representatives can only see the last 4 digits of a customer's national identity number), developers (who need real production data to test new software releases but should not be able to see sensitive financial data), vendors, etc.

Data erasure is a method of software-based overwriting that completely destroys all electronic data residing on a hard drive or other digital media to ensure that no sensitive data is leaked when an asset is retired or reused.

Exercise 34. *Find in text 3 the English for:*

засіб захисту даних від псування; забезпечувати конфіденційність; шифрування диску, апаратні механізми захисту даних та резервні копії; маскування та стирання даних; називатися оперативним шифруванням або прозорим шифруванням; запобігати крадіжці даних; псувати дані для того, щоби зробити їх невідновлюваними або непридатними для використання; забезпечувати потужний захист від зламу та несанкціонованого доступу; дозволяти доступ у разі підімкнення маркеру доступу та введення правильного ПІН-коду; використовувати захисний ключ-заглушку; реєструватися, виходити із системи та встановлювати різні рівні доступу; зчитуватися конт-

ролерами периферійних пристроїв; операційні системи, вразливі до зловмисних дій; виконувати несанкціоновані привілейовані операції; цілковито захищена система; забезпечувати відновлення втрачених даних; процес приховування (маскування) даних; маскування інформації від користувачів, розробників та постачальників; руйнувати дані, що містяться на жорсткому диску; запобігати витоку конфіденційної інформації; видаляти або повторно використовувати ресурс.

Exercise 35. *Say whether the statements below are true or false. Correct the false ones.*

1. Data security is the means of ensuring that data is kept safe from corruption and that access to it is suitably controlled. 2. Data security helps to ensure privacy and to protect personal data. 3. The technologies of software security are: disk encryption, hardware-based mechanisms for protecting data, backups, data masking and data erasure. 4. The technology of encrypting data on a key drive is often referred to as on-the-fly encryption or transparent encryption. 5. Software-based security solutions encrypt the data to prevent data from being corrupted. 6. A malicious program or a hacker may corrupt the data in order to make it unrecoverable or unusable. 7. Software-based security solutions can prevent access to data and hence offer very strong protection against tampering and unauthorized access. 8. Dongles can be used by anyone who can gain physical access to the system. 9. The software-based security uses a device that allows a user to log in, log out and to set different privilege levels by doing manual actions. 10. Hardware-based access control is more secure than protection provided by the operating systems as operating systems are vulnerable to malicious attacks by viruses and hackers. 11. The data on hard disks can be corrupted after a malicious access is obtained. 12. With hardware-based protection, it is impossible for a hacker or a malicious program to gain access to secure data or perform unauthorized privileged operations. 13. A completely secure system can be created using a combination of hardware-based security and secure system administration policies. 14. Backups are used to ensure recovery of the data which has been rewritten. 15. Data masking is the process of obscuring (masking) specific data within a database table or cell to ensure that data security is maintained and

sensitive information is not exposed to unauthorized personnel. 16. Data erasure is a method of software-based overwriting that completely destroys all electronic data residing on a hard drive or other digital media to ensure that no sensitive data is leaked when an asset is retired or reused.

Exercise 36. *Form all possible word combinations with the words from both columns. Translate them.*

1) to be	a) being stolen
2) can be used for	b) on-the-fly encryption
3) to offer	c) data
4) to be referred to as	d) suitably controlled
5) to enter	e) an alternative
6) to prevent from	f) this purpose
7) to corrupt	g) physical access
8) in order to	h) the correct PIN
9) to protect	i) be compromised
10) to gain	j) the operating system image

Exercise 37. *Fill in the blanks with prepositions* ***on, for, to, by, in, of, from, against, out*** *where necessary:*

1. Data security is the means … ensuring that data is kept safe … corruption and that access … it is suitably controlled. 2. The technologies … data security are: disk encryption, hardware-based mechanisms … protecting data, backups, data masking and data erasure. 3. Disk encryption is the technology that encrypts data … a hard disk drive. 4. Software-based security solutions encrypt the data to prevent data … being stolen. 5. Encrypted operating systems can be corrupted … a malicious program or a hacker, making the system unusable. 6. Hardware-based security solutions can prevent access … data and hence offer very strong protection … tampering and unauthorized access. 7. Newer technologies … hardware-based security solve this problem offering proof security … data. 8. The device uses biometric technology to prevent malicious users … logging … , logging … , and changing privilege levels. 9. The current state … a user … the device is read … controllers ... peripheral devices such as hard disks. 10. Hardware-based access control is more secure than protection

provided … the operating systems as operating systems are vulnerable … malicious attacks … viruses and hackers. 11. The data … hard disks can be corrupted … a malicious access.

Exercise 38. *Fill in the blanks with proper terms(**data erasure, backups, data masking, hardware-based security solutions, disk encryption, data security, software-based security solutions**) to complete the sentences.*

1. _____can prevent access to data and hence offer very strong protection against tampering and unauthorized access. 2. _____is the means of ensuring that data is kept safe from corruption and that access to it is suitably controlled. 3. _____ encrypt the data to prevent data from being stolen. 4. _____is the technology that encrypts data on a hard disk drive. 5_____ is a method of software-based overwriting that completely destroys all electronic data residing on a hard drive or other digital media to ensure that no sensitive data is leaked when an asset is retired or reused. 6. _____is the process of obscuring (masking) specific data within a database table or cell to ensure that data security is maintained and sensitive information is not exposed to unauthorized personnel. 7. _____are used to ensure recovery of the data which has been lost.

Exercise 39. *Answer the questions on text 3.*

1. What is data security? 2. What does data security help in? 3. What are the technologies of data security? 4. What is disk encryption? 5. What is the difference between software-based security solutions and hardware-based security solutions? 6. Why may security tokens be more secure? 7. When is access enabled, provided that security tokens are used? 8. What does hardware-based security device allow a user to do? 9. What technology does this device use? 10. What is the operating principle of hardware-based security? 11. Why is hardware-based access control more secure than protection provided by the operating systems? 12. How can a completely secure system be created? 13. What are backups used for? 14. What is data masking? 15. What may data be masked from? 16. What is data erasure?

Exercise 40. *Put all possible questions to the sentences below.*

1. Data security helps to ensure privacy. 2. The technologies of data security are: disk encryption, hardware based mechanisms for protecting data, backups, data masking and data erasure. 3. Disk encryption software or disk encryption hardware can be used for this purpose. 4. This technology is often referred to as on-the-fly encryption or transparent encryption. 5. A malicious program or a hacker may corrupt the data in order to make it unrecoverable or unusable. 6. Encrypted operating systems can be corrupted by a malicious program or a hacker, making the system unusable. 7. Hardware based computer security offers an alternative to software-only computer security. 8. The device uses biometric technology to prevent malicious users from logging in. 9. The current state of a user of the device is read by controllers in peripheral devices such as hard disks. 10. The hardware protects the operating system image and file system privileges from being tampered. 11. Backups are used to ensure recovery of the data which has been lost.

Exercise 41. *Translate into English.*

1. Захист даних – це засіб забезпечення захищеності даних від псування та спосіб належного контролю доступу до них. 2. Таким чином, захист даних допомагає забезпечити конфіденційність. 3. Технологіями безпеки даних є шифрування диску, апаратні механізми захисту інформації, резервні копії, маскування та стирання даних. 4. Шифрування диску – це технологія шифрування даних на жорсткому диску. 5. Цю технологію часто називають оперативним або прозорим шифруванням. 6. Шкідлива програма або хакер можуть зіпсувати дані та операційні системи, зробити їх невідновлюваними або непридатними до використання. 7. Рішення про забезпечення захисту апаратними засобами можуть сприяти запобіганню доступу до інформації, і тому є дуже потужним захистом від зламу та несанкціонованого доступу. 8. Доступ дозволений лише в разі підімкнення маркеру доступу і введення правильного ПІН-коду. 9. Проте той, хто отримує фізичний доступ до системи, може використати захисні ключі-заглушки. 10. Апаратний захист полягає у використанні пристрою, що дозволяє користувачеві реєструватися, виходити із системи та встановлювати різні рівні доступу. 11. Поточний стан користувача зчитується контролерами в периферійних пристроях, наприклад у жорстких дисках.

12. Нелегальний доступ зловмисника або шкідливої програми переривається контролерами жорсткого диску і DVD. 13. Апаратний контроль доступу є надійнішим за захист, який забезпечують операційні системи, тому що вони є вразливими до зловмисних дій вірусів та хакерів. 14. Апаратний захист унеможливлює отримання доступу до захищеної інформації або виконання несанкціонованих привілейованих операцій хакерами або шкідливими програмами. 15. Апаратні засоби захищають операційну систему та привілеї файлової системи від псування. 16. Отже, цілковито захищену систему можна створити з допомогою поєднання апаратного захисту та надійних заходів системного адміністрування. 17. Резервні копії використовують для забезпечення відновлення втрачених даних. 18. Маскування даних – це процес приховування конкретної інформації у таблиці або комірці бази даних для забезпечення підтримки захисту даних та приховування конфіденційної інформації від стороннього персоналу. 19. Воно може включати маскування інформації від користувачів, розробників та постачальників. 20. Стирання даних – це метод переписування, що цілковито руйнує інформацію, яка міститься на жорсткому диску або інших цифрових носіях, щоб унеможливити витік важливої інформації під час видалення або повторного використання ресурсу.

Exercise 42. *Compose a dialog on data security.*

Exercise 43. *Read, translate and write a summary of text 4.*

Text 4. Data Masking

Data masking is the process of obscuring (masking) specific data elements within data stores. It ensures that sensitive data is replaced with realistic but not real data. The goal is that sensitive customer information is not available outside of the authorized environment. Data masking is typically done while provisioning non-production environments so that copies created to support test and development processes are not exposing sensitive information and thus avoiding risks of leaking. Masking algorithms are designed to be repeatable so referential integrity is maintained.

Common business applications require constant patch and upgrade cycles and require that 6-8 copies of the application and data be made for testing. While organizations typically have strict controls on

production systems, data security in non-production instances is often left up to trusting the employee, with potentially disastrous results.

Creating test and development copies in an automated process reduces the exposure of sensitive data. Database layout often changes, it is useful to maintain a list of sensitive columns without rewriting application code. Data masking is an effective strategy in reducing the risk of data exposure from inside and outside of an organization and should be considered a best practice for curing non-production databases.

Effective data masking requires data to be altered in a way that the actual values cannot be determined or reengineered, functional appearance is maintained, so effective testing is possible. Data can be encrypted and decrypted, relational integrity is maintained, security polices can be established and separation of duties between security and administration established. Common methods of data masking include: encryption/decryption, masking and substitution.

Exercise 44. *Speak on effective data masking techniques.*

SUPPLEMENTARY READING

Text 1. Class Model

Class diagrams are fundamental to object-oriented analysis and design. The Unified Modelling Language (UML) is the industry standard notation for class diagrams. The class model shows static class objects in a system and the relationships between them. Two particularly important relationships are generalization (inheritance) and aggregation (whole-part). Each class object on the diagram often shows the class name, its attributes and operations. Details like data types for attributes and arguments for operations can also be shown on the diagram for some notations. Many class modelling notations are available but most developers have standardized on UML as illustrated below (Fig. 1).

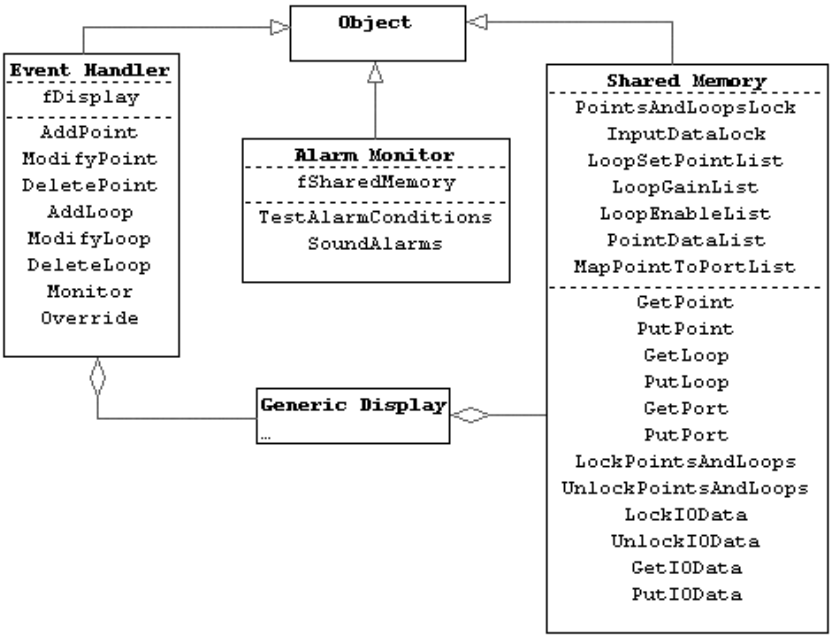

Fig. 1. UML Class Model

Notes

Notation – нотація, система позначень, набір символів і правил для запису синтаксису

generalization – узагальнення
inheritance – 1) успадкування; 2) спадщина; 3) спадок
aggregation – агрегування *(механізм багаторазового використання одним об'єктом методів іншого об'єкта; цей механізм реалізує розподіл інтерфейсів)*
event handler – обробник подій
point – пункт
point data – координати точок, дані про координати
map point – план / схема пунктів / координат
override – 1) перевизначення; 2) обхід; 3) скасування *(команди)*
alarm monitor – монітор аварійних сигналів
alarm condition – аварійна / небезпечна ситуація
sound alarm – звукове попередження про небезпеку
shared memory – розподілена пам'ять
generic – типовий, стандартний, звичайний
lock – замикати, блокувати
unlock – 1) розмикати, розблоковувати; 2) розблокування
loop gain – коефіцієнт підсилення замкненого ланцюга
enable – 1) активувати, вмикати; 2) розблокувати; 3) уможливлювати, дозволяти
port – порт

Assignments

1. Read and translate text 1.
2. Find in the text the English for:

показувати статичні об'єкти класу та важливі відношення між ними; узагальнення (наслідування) та агрегування; показувати назву класу, його атрибути та операції; атрибути та аргументи для операцій; система позначень у моделюванні класів; обробник подій; монітор аварійних сигналів; звукове попередження про небезпеку; небезпечна ситуація; розподілена пам'ять.

3. Ask questions on the text.
4. Use Fig.1 to describe a UML class model.

Text 2. Data Model

An entity-relation diagram, called an ERD, illustrates the data structure of an information system. A database can be designed using logical and physical data models that highlight primary and foreign keys. Martin's Information Engineering notation is typically used for data models. An entity-relation diagram (ERD), also referred to as a data model, typically shows the name of each entity, its list of attributes and relation-

ships with other entities. Primary keys can be identified for each entity and foreign keys can be used to express relationships between entities.

Often the data model is later implemented by generating Structured Query Language (SQL) for a Relational Database Management System (RDBMS) where entities become tables and attributes become columns. Martin's Information Engineering notation is illustrated in Fig. 2.

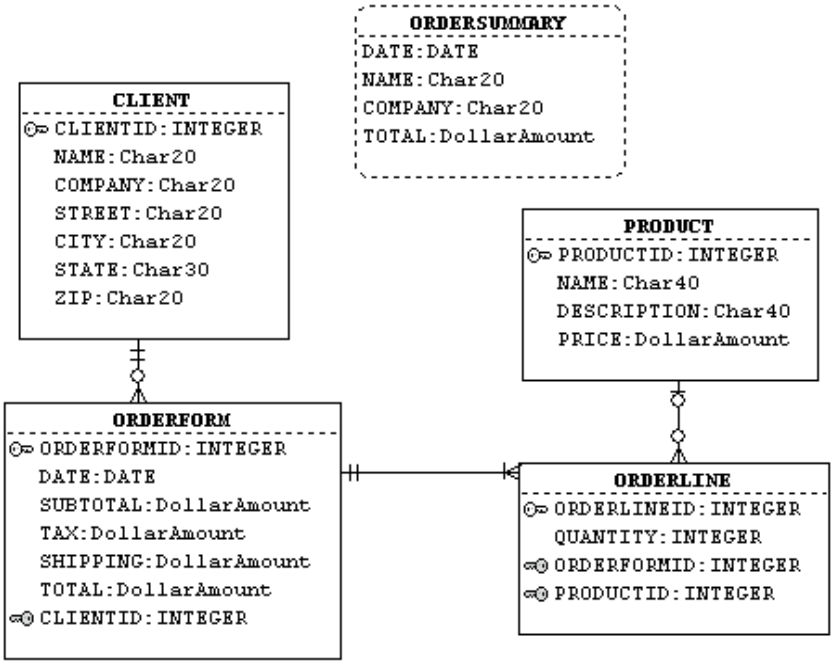

Fig. 2. Information Engineering ERD

Notes

Data model – модель даних

entity-relation diagram (ERD) – діаграма відношень логічних об'єктів-сутностей, ER-діаграма

highlight – підкреслювати, виділяти, акцентувати увагу

primary key – первинний ключ *(у базах даних)*

foreign key – зовнішній ключ *(у базах даних)*

information engineering – інформаційна інженерія, інфотехніка

entity – (*логічний*) об'єкт

Structured Query Language (SQL) – мова структурованих запитів, мова SQL

Relational Database Management System (RDBMS) – реляційна система керування базою даних, СКРБД *(інформацію в таких базах даних зберігають у двовимірних таблицях, які називають відношеннями (relations); кожний стовпчик таблиці має назву «атрибут» (attribute), який описує тип елементів стовпчика; рядок даних таблиці називають кортежем (tuple)*

dollar amount – сума в доларах

ID (identifier) – ідентифікатор

char (character) – знак

ZIP – найбільш поширений стандарт ущільнення; формат архівів на FTP (File Transfer Protocol)-серверах

integer – ціле число

shipping – поставка, відправлення, відвантаження

orderline – кількість найменувань у замовленні

Assignments

1. Read and translate text 2.
2. Find in the text the English for:

модель даних; діаграма відношень логічних об'єктів; виділяти первинні та зовнішні ключі; список атрибутів та зв'язків з іншими логічними об'єктами; мова структурованих запитів; реляційна система керування базою даних (СКРБД); підсумок; сума в доларах; ціле число; ідентифікатор.

3. Ask questions on the text.
4. Use Fig.2 to describe Information Engineering ERD.

Text 3. Process Model

The process model is typically used in structured analysis and design methods. Also called a data flow diagram (DFD), it shows the flow of information through a system. Each process transforms inputs into outputs.

The Gane & Sarson style DFD shown in Fig. 3 is typically used for information systems. The diagram shows the flow of information in a small software company.

Notes

Data flow diagram – діаграма потоків даних

context diagram – контекстна діаграма *(модель, яка зображає систему з її оточенням на високому рівні абстрагування, – вона містить об'єкти, які є зовнішніми щодо системи і взаємодіють з нею, але вона не показує внутрішню структуру системи та її поведінку)*

explode – *тут*: розбиватися *(на складові)*
parent diagram – породжувальна діаграма, діаграма першого рівня
child diagram – породжена діаграма, діаграма другого рівня
specification – специфікація
packet queue – черга пакетів
trigger – пусковий сигнал
key entry – клавішне введення
receiver – приймач
PAT = PaT= P&T – 1) (picture and text) – «малюнок і текст», режим або функція PAT 2) program association table – таблиця з перерахуванням програм потоку та їх ідентифікація
match – відповідність, збіг
sales income – прибуток від реалізації
invoice – рахунок-фактура
inventory – інвентаризація, облік товару

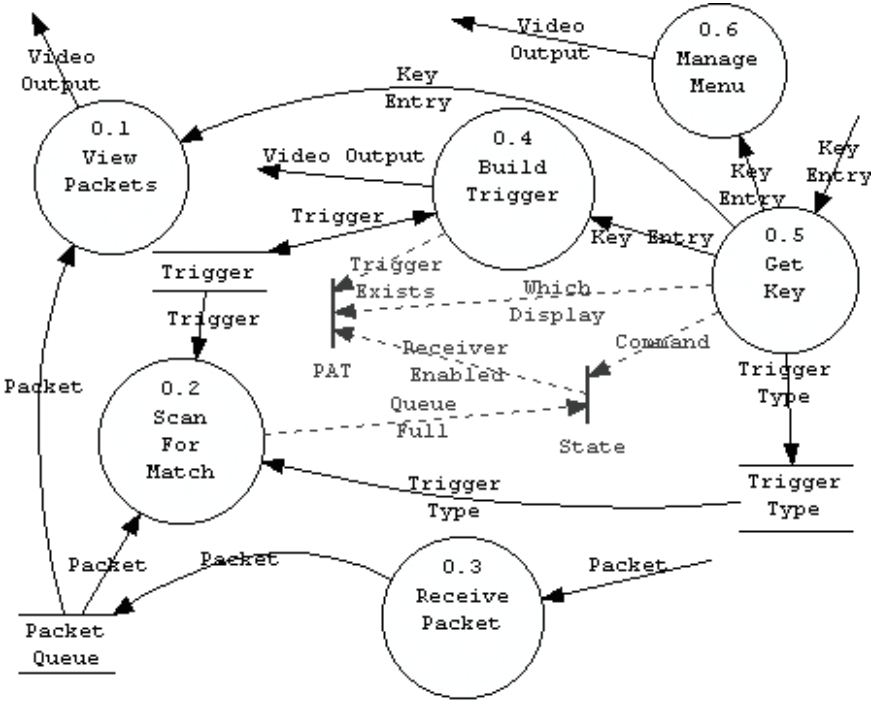

Fig. 3. Yourdon/DeMarco DFD

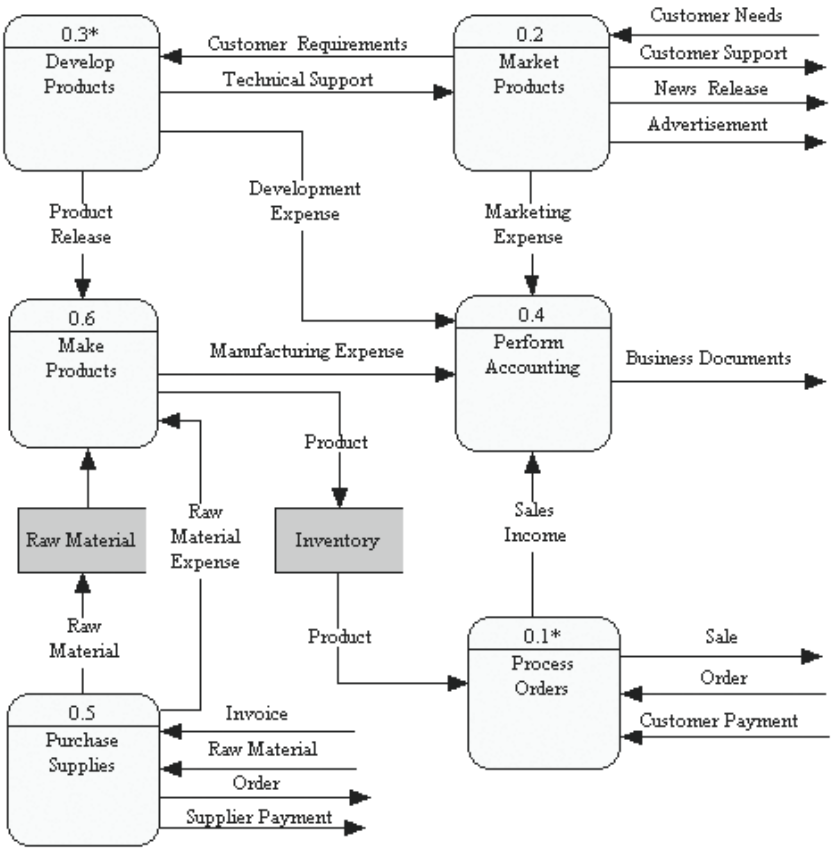

Fig. 4. Gane & Sarson DFD

Assignments

1. Read and translate text 3.
2. Find in the text the English for:

діаграма потоків даних; методи структурного аналізу та проектування; зовнішні об'єкти; за межами системи; діаграми першого та другого рівнів; проектування та аналіз систем у реальному часі; текстова специфікація; відеовихід; черга пакетів; клавішне введення; активувати; потреби клієнтів; підтримка споживачів; реклама; витрати на розроблення; технічна підтримка; випуск продукції; проведення бухгалтерського обліку; сировина; опрацьовувати за-

мовлення; ділові документи; прибуток від реалізації; рахунок-фактура; облік товару.
3. Ask questions on the text.
4. Describe Yourdon/DeMarco and Gane & Sarson DFDs using fig. 3 and 4.

Text 4. State Models

The state model describes the states and events in a system using a diagram or table. There are many different types of state diagrams and tables. Causal loop diagrams are used for system models.

In a structured analysis and design method, state models show the modes in a system and usually connect to data flow diagrams using control bars and control flows. In an object-oriented approach a state model is typically used to describe the lifecycle of a complex object. The Harel state model illustrated in Fig. 5 has become popular in recent years due to its ability to express concurrency.

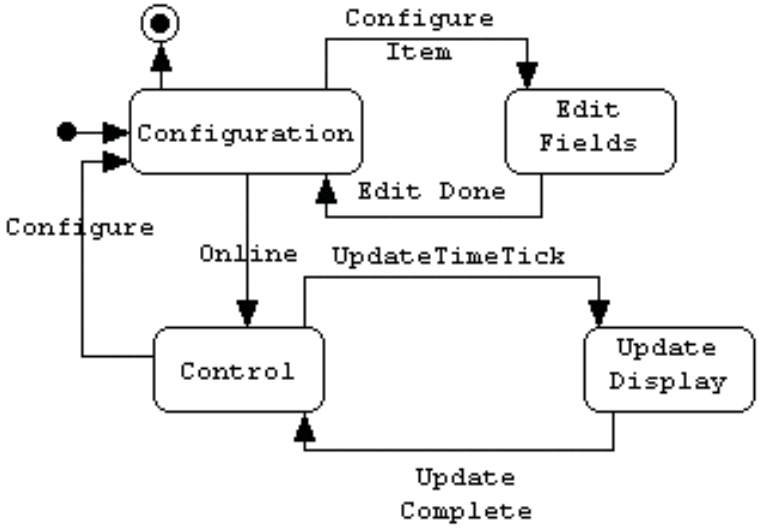

Fig. 5. The Harel state model

Tables are also an effective way of expressing information about states and events as illustrated in the state transition table below. Tables can be created with the table editor much like filling in cells of a spreadsheet. Tables can also be generated automatically from information in a state diagram.

STATE	EVENT	ACTION	NEXT STATE
Wait For Dollar	Bill Detected	Load Bill	Verify Dollar
Verify Dollar	Verification Failed	Reject Bill	Wait For Dollar
Verify Dollar	Verification Passed	Dispense Coins	Dispensing Coins
Dispensing Coins	Sufficient Funds Remain	Accept Another Dollar	Wait For Dollar
Dispensing Coins	Insufficient Funds Remain	Turn On Out Of Money Light	Out Of Money
Out Of Money	Money Refill	Accept Another Dollar	Wait For Dollar

Fig. 6. State transition table

Notes
State model – модель станів
causal – причиновий
loop diagram – контурна схема
bar – прямокутник *(на блок-схемі)*
control flow – потік керування
concurrency – паралелізм
update time – час поновлення даних
tick – імпульс
state transition – перехід станів
bill – банкнота, купюра, рахунок
reject – не приймати, відкидати
dispense – видавати
coin – монета
sufficient – достатній
funds – гроші, кошти, сума
refill – 1) поповнення; 2) поповнювати

Assignments

1. Read and translate text 4.
2. Find in the text the English for:
описувати стани та події; метод структурного аналізу та проектування; об'єктно-орієнтований підхід; через свою здатність передавати паралелізм; імпульс часу поновлених даних; ефективний спосіб представлення інформації; таблиця переходу станів; заповнення комірок електронної таблиці; завантажувати купюру; перевіряти долари; видавати монети; достатня сума; недостатня сума; засвітити табло «немає грошей»; поповнювати кошти.
3. Ask questions on the text.
4. Use Fig. 5. to describe the Harel state model.

Text 5. Object Models

An object model shows object instances, their operations and messages between objects to document the design mechanisms within an object-oriented design. Each diagram illustrates a part of the design with a collection of communicating objects.

Popular notations include UML sequence diagrams and UML collaboration diagrams.

UML communication (called collaboration prior to UML 2.0) diagram is a type of object interaction diagram that emphasizes the data links between participants. It is illustrated in Fig. 7 below with association lines drawn between objects and couples attached to the lines showing the name, direction and parameters of object operation calls.

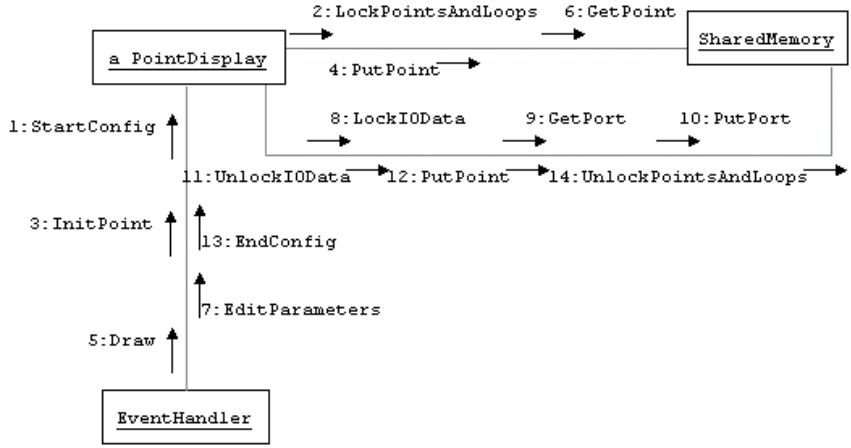

Fig. 7. UML Communication Diagram

The UML Sequence diagram (Fig. 8) shows the same type of information, but it highlights the time sequence of object operations and events.

Assignments

1. Read and translate text 5.
2. Answer the questions on the text.
   - What does an object model show?
   - What is UML communication?
   - What do the couples attached to the lines in UML communication diagram show?

- What does the UML sequence diagram highlight?

3. Describe a UML Communication Diagram and UML Sequence Diagram using Fig. 7 and 8.

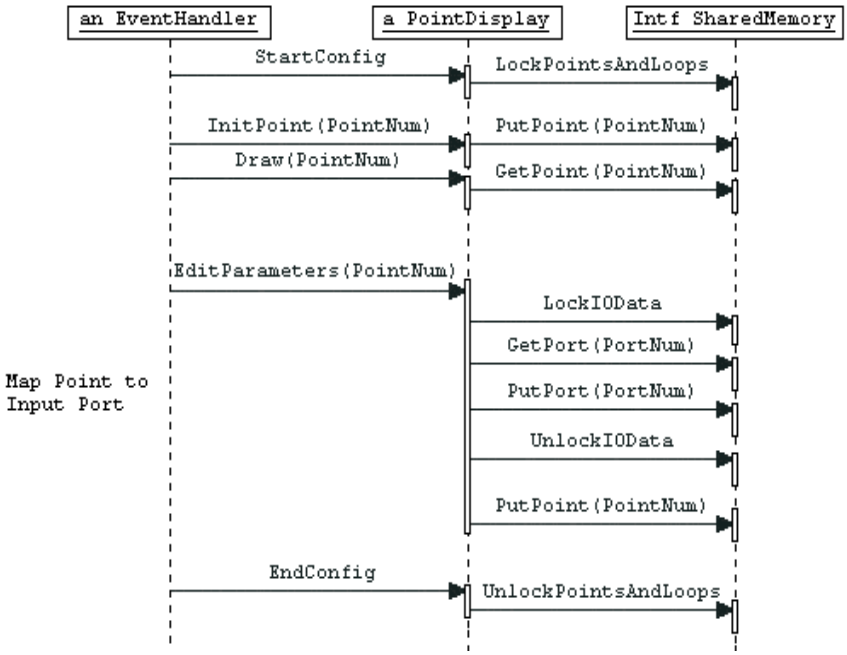

Fig. 8. UML Sequence Diagram

Text 6. Task Model

The task model is used to create high level system or architectural diagrams of a software system or lower level task diagrams showing threads of execution.

Architectural diagrams (Fig. 9) illustrate the physical design of complex systems using processors, devices and interfaces. Client-server or other distributed systems can be shown with predefined or user defined icons like computers, printers or communication systems.

Task diagrams (Fig. 10) show detailed interactions between independent threads of execution (tasks or interrupt service routines). These interactions use operating system services including queues, semaphores, mailboxes, event flags and input/output ports.

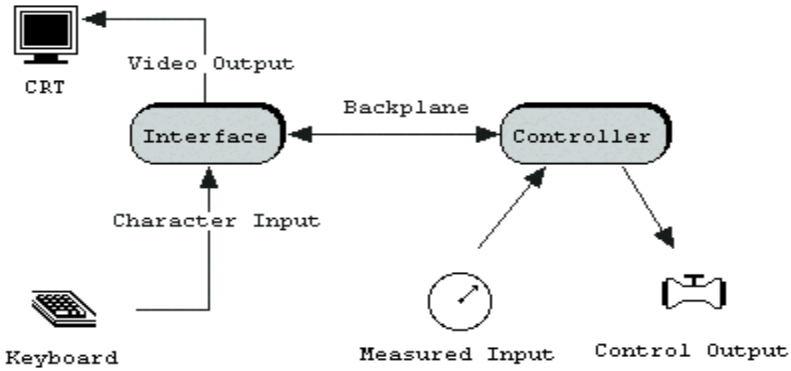

Fig. 9. System (Architectural) Diagram

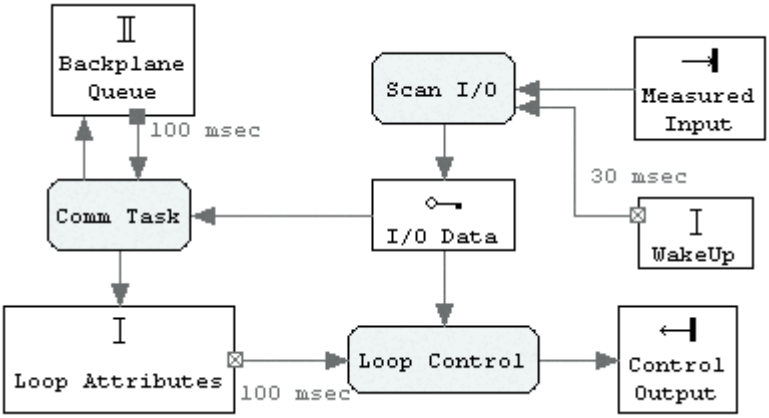

Fig. 10. Task Diagram

Notes
Thread – потік, тред
task model – модель завдання
backplane – з'єднувальна плата
interrupt service routine – підпрограма оброблення переривання
semaphore – семафор
event flag – прапорець події
msec (millisecond) – мілісекунда
wakeup – активізація

Assignments

1. Read and translate text 6.
2. Find in the text the English for:
архітектурні діаграми; потоки виконання; діаграма системи; розподілені системи; попередньо визначені піктограми; діаграма завдання; з'єднувальна плата; програма оброблення переривання; прапорець події; виміряний вхід; активізація.
3. Answer the questions.
   - What is the task model used for?
   - What do architectural diagrams illustrate?
   - What can client-server or other distributed systems be shown with?
   - What do task diagrams show?
   - What do the interactions between independent threads of execution use?
4. Describe a System (Architectural) Diagram and a Task Diagram using fig. 9 and 10.

Text 7. Software Development

In today's software world, there are boundless discussions and thoughts around "how best to develop software." There are many books and standards written on this subject. However, it all comes down to the organization, the expectations of the users/business, and how much time and money is provided to software projects.

The Integrated Development Environment (IDE) directs an organization toward specific tools as well as techniques of modelling.

If an organization requires documentation, there are several possible approaches to design and modelling techniques. Most of the expectations in any development organization are focused on creating the code. While design is an absolute necessity, the method of describing the design varies greatly, as does the techniques used to produce such designs. The level of detail will often dictate the effort and time required to produce the artifacts that communicate the design of the system.

Technology offers some great capabilities in pulling together the design of any software system. Regardless of platforms, the technologies available to create models are vast and highly efficient in today's development world. Design techniques include standard modelling languages such as the Unified Modelling Language (UML), frameworks such as the Model-Driven Architecture (MDA), and software processes

such as the Enterprise Unified Process (EUP). Tools are available for download, purchase from various vendors, and even developed in-house. Modelling tools that support the standards should be the first choice for organizations. The Integrated Development Environment (IDE) directs an organization toward specific tools as well as techniques of modelling.

In most discussions of "how" modelling is accomplished, the conversation will ultimately include topics such as diagrams, artifacts, technologies, and platforms. While this certainly does not sum up modelling as a phase of development, it is often the primary considerations. While having these discussions will guide the organization towards appropriate decisions of "how to design," you must also consider standards such as the Software Engineering Body of Knowledge (SWEBOK), IEEE 1074-2006, and ISO 9001 for appropriate design techniques and concerns.

Assignments

1. Read and translate text 7.
2. Ask questions on the text.
3. Decipher the abbreviations: MDA, ISO, UML, IDE, SWEBOK, EUP
4. Divide the text into logical parts and entitle each of them.

Text 8. ModellingTechniques

Modelling techniques usually come down to artifacts (diagrams). You must define the desired artifacts for the system definition, the audience for the artifacts, and how the artifact will be used. If documentation is the goal, then technology and techniques should work towards delivering structured output, be that in the form of a document or a graphics representation of the design (diagrams). The goal should be to produce only what is needed and what is effective to communicate the design. You should not produce content or artifacts "just because you can." Reduce output for the system specification to pertinent details, focusing more information on software modelling and UML.

If you want to effectively communicate the architect's design to the developers producing the code, then technology that closely ties design techniques with the resulting code should be your focus. Little attention to documentation output and large focus on effective communication

will help produce desired project results. People and processes define effectiveness for development results while technology defines the efficiency of that same effort. Design is the software development life cycle (SDLC) step between the customer needs and the developer code. Make that step as streamlined and effective as possible while being efficient on the part of all interested parties. If customers don't need to see it, discuss it, or approve it, then do not consume their time by producing it. If developers spend unnecessary cycles translating documents to code, give them efficiency by sharing design technologies in the coding phase. The techniques that accomplish bringing together requirements, design and code should be the focus for improving the development effort. This is true whether waterfall or agile methods are the software development process for the organization.

Assignments

1. Read and translate text 8.
2. Ask questions on the text.
3. Speak on software design and modelling.

Text 9. Holographic Storage

Holographic storage has been a dream of scientists for 30 years. "Many of the ideas are nearly as old as hard disk magnetic recording," said Glenn Sincerbox, a data storage expert for the International Business Machines Corporation, who worked on holographic memory in the 1960's. But while magnetic recording has blossomed into a huge industry, holographic storage has never emerged from the laboratory. But nowadays the experimental memory device stores information using holograms, those shimmering three-dimensional images that are commonly seen on credit cards, and each person has opportunity to use in everyday life. But the creation of these cards required great efforts of Microelectronics and Computer Technology Corporation scientists (a computer-industry consortium usually known by the initials M.C.T.C). While there are still formidable obstacles, M.C.T.C. researchers are confident that the holostore, as they call it, will be available later this decade.

Holographic storage differs from disk drives in three respects:

1. Disk drives store information in only two dimensions, on the flat surface of the disk. Holographic systems store information in three dimensions inside a cube of special material, allowing packing more information into a given space.

2. Disk drives have moving parts. It can take many thousandths of a second – an eon for computers – for the recording head to reach the spot on the spinning disk where the desired data is stored. But holographic systems can be built with no moving parts and, using lasers, can find the needed data in a millionth of a second.

3. Perhaps most important, disk drives store or retrieve a single bit – a zero or one of computer code – at a time. Holographic systems store or retrieve an image at a time, and each image can consist of thousands or even millions of bits.

The computer holograms consist of a pattern of black-and-white squares that represent zeros and ones. Each pattern is known as a page, which can consist of millions of bits, and as many as 100,000 pages might be stored in a single crystal.

The intersection of two beams creates an interference pattern of light and dark areas, which in turn changes the light-bending characteristics of the crystal.

To retrieve the data, the reference beam alone is shined through the crystal. Its light is bent in such a way that the original image of the page is reconstructed. Pointing the reference beam at different angles retrieves different pages.

Some experts say holographic and other futuristic storage systems may never catch up with disk drives or with semiconductor chips, both of which continue to advance in storage capacity and to become cheaper. To overcome the slow speed at which disk drives transfer data, companies are starting to use arrays of drives working in parallel.

But excitement about holographic storage has picked up recently as several significant developments have been reported. Bellcore researchers have developed a chip that contains 10,000 microscopic lasers. Each laser points to the crystal at a different angle, so retrieving a particular page requires turning on the correct laser, which is faster than trying to steer a single reference beam.

Holography is not the only thing that demands the specific attention. With Pentagon funding, Peter M. Rentzepis, professor of chemistry at the University of California at Irvine, is working on a similar three-dimensional storage system using a different principle. In his system, two laser beams are sent into a storage cube at right angles to each other. The cube is made of plastic laced with an organic chemical that has the property of allowing single laser beams of a particular frequency

to pass through. But at the point where two beams cross, their combined energy is such that they are absorbed, producing a change in the material. Changed spots can be used to represent ones in the digital code; unchanged spots, zeros.

Assignments

1. Read and translate text 9.
2. Ask questions on the text.
3. Write a summary of the text "Holographic storage".

Text 10. Storage Structures

Databases may store relational tables/indexes in memory or on hard disk in one of many forms:

- ordered/unordered flat files
- ISAM (Indexed Sequential Access Method)
- heaps
- hash buckets
- logically-blocked files
- B+ trees

The most commonly used are B+ trees and ISAM.

Object databases use a range of storage mechanisms. Some use virtual memory-mapped files to make the native language (C++, Java etc.) objects persistent. This can be highly efficient but it can make multi-language access more difficult. Others disassemble objects into fixed- and varying-length components that are then clustered in fixed sized blocks on disk and reassembled into the appropriate format on either the client or server address space. Another popular technique involves storing the objects in tuples (much like a relational database) which the database server then reassembles into objects for the client. Other techniques include clustering by category (such as grouping data by month, or location), storing pre-computed query results, known as materialized views, partitioning data by range (e.g., a data range) or by hash.

Memory management and storage topology can be important design choices for database designers as well. Just as normalization is used to reduce storage requirements and improve database designs, conversely denormalization is often used to reduce join complexity and reduce query execution time.

Indexing

Indexing is a technique for improving database performance. The many types of index share the common property that they eliminate the need to examine every entry when running a query. In large databases, this can reduce query time/cost by orders of magnitude. The simplest form of index is a sorted list of values that can be searched using a binary search with an adjacent reference to the location of the entry, analogous to the index in the back of a book. The same data can have multiple indexes (an employee database could be indexed by last name and hire date.)

Indexes affect performance, but not results. Database designers can add or remove indexes without changing application logic, reducing maintenance costs as the database grows and database usage evolves.

Given a particular query, the DBMS' query optimizer is responsible for devising the most efficient strategy for finding matching data. The optimizer decides which index or indexes to use, how to combine data from different parts of the database, how to provide data in the order requested, etc.

Indexes can speed up data access, but they consume space in the database, and must be updated each time the data are altered. Indexes therefore can speed data access but slow data maintenance. These two properties determine whether a given index is worth the cost.

Transactions

Most DBMS provide some form of support for transactions, which allow multiple data items to be updated in a consistent fashion, such that updates that are part of a transaction succeed or fail in unison. The following rules characterize this behaviour:

- Atomicity: Either all the data changes in a transaction must happen, or none of them. The transaction must be completed, or else it must be undone (rolled back).
- Consistency: Every transaction must preserve the declared consistency rules for the database.
- Isolation: Two concurrent transactions cannot interfere with one another. Intermediate results within one transaction must remain invisible to other transactions. The most extreme form of isolation is serializability, meaning that transactions that take place concurrently could instead be performed in some series, without affecting the ultimate result.

- Durability: Completed transactions cannot be aborted later or their results discarded. They must persist through (for instance) DBMS restarts.

In practice, many DBMSs allow the selective relaxation of these rules to balance perfect behaviour with optimum performance.

Replication

Database replication involves maintaining multiple copies of a database on different computers, to allow more users to access it, or to allow a secondary site to immediately take over if the primary site stops working. Some DBMS piggyback replication on top of their transaction logging facility, applying the primary's log to the secondary in near real-time. Database clustering is a related concept for handling larger databases and user communities by employing a cluster of multiple computers to host a single database that can use replication as part of its approach.

Assignments

1. Read and translate text 10.
2. Ask key questions on the text.
3. Make up a plan of the text.

Text 11. Effective Use of Architecture

Effective use of architecture is indicated by the creation of a plan, addressing critical system-wide issues, for structuring the resulting system and then developing the system in compliance with that plan. Thus, most effective architectures are defined prior to detailed development and constrain detailed design. They carry the decisions made by the architect and disseminate those decisions to developers.

There must be clear identification of structural patterns and coordination patterns that abstract important characteristics of the application domain. The architects must clearly identify a small number of key patterns of structure that are sufficient for implementation of the system. These structural patterns are templates for the organization of the system and elaboration of its components. They must be clear and understandable with minimal complexity and the number of patterns must be relatively modest. Each pattern, to be effective, must leverage a substantial portion of the system organizational structure. They serve to reduce the

variability of design across a system while clearly identifying the anticipated locations for variation. The reduction of variability minimizes the complexity of the system allowing more effective reasoning about the system as a whole based on the understanding of the characteristics of a small number of repeating patterns. They also facilitate the standardization of infrastructure services and common utilities and provide a consistent context for components that increases component reusability. The notion of leveraging system understanding from the characteristics and properties of a small number of archetypes applies equally to coordination of activities in the system. These coordination patterns or protocols characterize the interactions that are permitted between components of the system and allow designers to leverage understanding of how the system components dynamically interact by understanding the properties and behaviour of a small number of coordination protocols. Each interface in the system is mechanized with one or more of these coordination protocols. The architecture usually embodies these coordination patterns in infrastructure capabilities that are made available for use by the various system components.

Architectures must provide identification of partitions within the system. A primary division is that between system infrastructure and the applications. The architecture must clearly identify the existence of a system infrastructure, defined in detail by the architecture specification, and the existence of the system applications, parts that implement the mission specific functionality of the system. Infrastructure is ubiquitous and provides capabilities to all applications uniformly. The architecture specification identifies the details of infrastructure design and characteristics, including the interfaces used by applications. This gives an unambiguous picture of what system capabilities are provided as common services to the application developers and allows them to focus on application related details and not be diverted by functionality required to integrate their components into the system. Further partitions are identified within the infrastructure to organize the system wide capabilities and allow efficient allocation of requirements related to common services and overall integration of the system. Applications partitions are also identified along with the allocation of functional requirements to them. This represents the first mechanism for validating the coverage of system requirements. Application partitions provide placeholders for mission specific functionality defined by the application developers.

Partitions are a way of ordering a system in its decomposition. There are a great many ways of organizing solutions to a design problem but partitions are not arbitrary. We generally expect tighter cohesion and coupling among components within the same partition than between components across partition boundaries. Partition boundaries are often a location where architectural constraints govern the interaction of components. Tighter cohesion between the functions allocated to a partition allows* for components within a partition that are more tightly focused on a specific set of application capabilities and helps reduce the number of interfaces required within partitions and between partitions. Partitions do not necessarily reflect the physical model of the system or structure of compilation units.

An architecture must identify interfaces between the target system and external systems and major and critical interfaces within the target system itself. Architectural views should be provided that illustrate the interfaces and identify those infrastructure capabilities (and the coordination patterns they implement) used to mechanize each interface. Architectural views for a reference architecture will identify the external and component level interfaces by specific details where known and by abstractions where they are determined by target system elaboration. Target architectures will provide specific details of interface content.

Assignments

1. Read and translate text 11.
2. Ask key questions on the text.
3. Give the gist of the text.

Text 12. Architectural Specification

The architectural specification should fully characterize interface mechanisms so that infrastructure developers will know what to build and application developers will know what capabilities are available and how to use them.

The architectural specification should identify common services and utilities, other than interface mechanisms, that the infrastructure will provide. The specification should describe how they are used, what their intent is, what assumptions are incorporated in their design, and the de-

* allow for – передбачати; враховувати, брати до уваги

tails of their interface. The architecture should identify system states and modes and describe what each component is expected to provide to support those states and modes and what general behaviours they can expect from the system in each state and mode.

The development environment in support of the architecture should provide structural types used for component construction. Structural types are a realization of structural patterns identified during architectural design and embody the coordination protocols supported by the infrastructure. They identify locations where developers must provide functionality and where development tools or templates provide default capabilities. Each component of the system is an instance of a structural type defined by the architecture. The environment may provide special tooling, component templates, or instruction guides for developers to use in instantiating these types.

The architectural design must provide an effective information protection strategy to encapsulate components, infrastructure, and external interfaces. This protection strategy supports a decoupling of components from the implementation details of other components and external systems. Within the constraints of performance requirements this is one of the most important properties enforced by an architecture. It ensures conceptual integrity by managing proliferation of complexity and allows abstraction of stereotypical behaviours. It is a major contributor to the ability of the architecture to provide intellectual control over the system. It insures against side effects and hidden interfaces. An effective information protection strategy has a major impact on the integrability and maintainability of the system as well as on its reliability. We look for protection from the specific details of system components such as operating systems, file management systems, database management systems, hardware devices, inter-process communications facilities, and user interfaces.

Architectures should isolate system components from the environment in which they operate as far as is feasible. Application components should not be tasked with dealing with contention for system resources, determining communications routing, or maintaining a model of the location of other system components, except where unavoidable.

The architecture should provide an explicit description of the assumptions built into the architecture including those required by the infrastructure. It should further provide a detailing of the assumptions that

components make that effect the interfaces in which they participate. Knowledge of the assumptions used in constructing system components is invaluable in being able to integrate the system in a timely and cost effective manner and in maintaining the system over its lifetime.

The architecture should support the incremental addition of system capability. The infrastructure should support incremental development allowing new functionality to be added without requiring changes to existing functionality or without requiring extensive retest of previously integrated components. Incremental development facilitates integration, parallel development of system components, and testing of components as black boxes. This reduces the need to retest components when changes are made to implementations or when common components are incorporated into other products.

The architecture should provide instructions and guidelines for detailed decomposition and lower level partitioning by developers during detailed design. These should be based upon a comprehensive approach to systematize the analysis of the system and documentation of the design. These instructions reflect choices about particular analysis and design paradigms and, frequently, reflect capabilities of the development support environment. It allows for compatible methods of analysis and design across the system that enable modellingof the system and parameterization of component costs (technical as well as financial).

Architectures should foster the creation of the simplest solution to a system problem that is consistent with performance requirements, functional requirements, and system qualities.

Assignments

1. Read and translate text 12.
2. Ask questions on the text.
3. Speak on characteristics of effective architecture.

Text 13. Effective Architectural Requirements

In general, an architecture should strive through its standards for uniformity across a single class of problem. Systems are more robust and more predictable when a uniform solution is applied across the system instead of using a collection of unique point solutions for what is essentially the same problem.

Adaptation instructions and implementation guidelines for the architecture should provide a clear identification of what decisions or kinds of decisions are left to the discretion of designers and what decisions are conveyed by structural types. Adaptation instructions and implementation guidelines identify where application designers must provide functionality to satisfy allocated requirements. They also identify where implementers may use discretion in providing unique or non-standard components or component implementations and interface mechanizations. They further identify what infrastructure capabilities must be used by components and what behaviour and service those capabilities provide components.

Exceptions to any architecture are probably inevitable, especially over an extended lifetime, and in particular where a reference architecture is instantiated for multiple target systems (products). There must be specific processes defined for how exceptions are requested and adjudicated including where the authority lies to grant exceptions. The processes must ensure that exceptions are granted only for compelling reasons since they are a source of complexity that can compromise conceptual integrity and intellectual control. They can also foster the spawning of multiple baselines offering relatively low payback and limited additional capability in the face of high maintenance costs. The processes must, however, be reasonable and efficient.

The architecture must anticipate the necessity for exceptions and provide mechanisms to accommodate them with a minimal degradation of system qualities. The architecture must be designed for change, and must acknowledge the need for variation. Where applicable, It must anticipate the use of legacy components or components from other vendors (e.g., COTS*) with foreign architectures in a way that preserves the conceptual integrity of the system.

The architecture must clearly derive from a set of driving architectural requirements that are adjudged by the architects and other stakeholders to be of sufficient importance to the system and its qualities that specific architectural mechanisms are required to insure their coverage. These requirements must be a manageably small set or the resulting architecture will be unwieldy and brittle. There is a fine balance between controlling development and over-controlling development and quash-

* COTS – (commercial off-the-shelf) готовий, що є в продажу; купований

ing innovation or forcing application developers to im-plement inefficient, overly complex, or fragile components. The architecture and its infrastructure should be the minimum required to guarantee satisfaction of the driving requirements. Requirements not critical to the system overall should be allocated to application components or utilities, as appropriate. Driving requirements should succinctly establish the need for each capability and capabilities not supported by driving requirements should be omitted.

The architecture must provide adequate documentation so all users can understand how to employ it and how it structures the system and its behaviours. There must be a sufficient number of static views and dynamic views. It should include a high level description sufficient to impart the abstractions provided by the architecture. It also should include sufficient details for understanding how infrastructure services and utilities operate, what specific capabilities they provide to components, the intent of the developers with regards to their use, and how they are to be used. Documentation should include adaptation guidelines for instructing architects and systems engineers how to adapt the architecture to products and how to instantiate target architectures from the reference architecture, where applicable. Guidelines for implementation must also be provided for component developers including a complete description of structural types and their use in creating components. Those guidelines must also discuss the coordination protocols that govern how the components communicate and synchronize so developers understand how to integrate multiple components. If architects anticipate the reuse of the architecture on multiple products they should provide information for systems designers on how to analyze their problem within the framework of the architecture and guidelines for evaluating the suitability of the architecture for their purposes.

Assignments

1. Read and translate text 13.
2. Ask questions on the text.
3. Divide the text into logical parts and entitle each of them.

Text 14. Examples of Multimedia

Multimedia is the use of several types of outputs from a computer – or other device – in order to give the user a ‘richer’ and more interesting

experience. Listening to a CD with earphones is not multimedia because you are only experiencing one media – sound. Video is a good example that uses colour, onscreen motion and sound.

Games on computers first started out without any sound at all, and many people thought that the inclusion of sound to a game would not make much difference. However it was soon found that adding even simple sounds enhanced the overall experience of the computer user. The use of multimedia also allows the possibility of user input to make selections and choices. This is called interactive multimedia, because the user interacts with the material in some way.

If you send a CD of digital photos of your party to a friend and add some sound clips, you are preparing a multimedia presentation because you are using more than one type of media. The user can also interact with the material e.g. stop the display, go to various places on the video, answer questions that are posed, select from various options that are presented by the interactive package, can ask for help or extra information.

A virtual reality application – e.g. a cockpit simulator that is used to train pilots on the ground – is an example of multimedia. In a virtual reality simulator the pilot, though is not actually in the plane, feels as though he is.

Assignments

1. Read and translate text 14.
2. Ask questions on the text.
3. Speak on the examples of multimedia.

Text 15. Multimedia Application

Multimedia finds its application in various areas including art, education, entertainment, engineering, medicine, mathematics, business and scientific research. In education multimedia is used to organize computer-based training courses (popularly called CBTs) and produce reference books like encyclopaedia and almanacs. A CBT lets the user go through a series of presentations, texts on particular topics and associated illustrations in various information formats. The Multimedia Messaging System, or MMS, is an application that allows one to send and receive messages containing multimedia – related content. MMS is a common feature of most cell phones. An electronic multimedia encyclopedia can present information in better ways than traditional

encyclopaedia, so the user has more fun and learns more quickly. For instance, an article on World War II can include hyperlinks to articles on countries involved in the war. When users click on a hyperlink, they are redirected to a detailed article about that country. In addition, it can include a video on the Pacific Campaign. It can also present maps pertinent to World War II. This can speed-up learning and improve the user experience, when added to multiple elements such as pictures, photographs, audio and video. (It is also said that some people learn better by seeing than reading and some others by listening).

Multimedia is heavily used in the entertainment industry, especially to develop special effects in movies and animation for cartoon characters. Multimedia games are a popular pastime and are software programs available either as CD-ROMs or online. Some video games also use multimedia features.Multimedia applications that allow users to actively participate instead of just sitting by as passive recipients of information are called *Interactive Multimedia*.

Assignments

1. Read and translate text 15.
2. Ask key questions on the text.
3. Write a summary of the text.

Text 16. Software Vulnerability to Attack

What makes it so easy for attackers to target software is the virtually guaranteed presence of vulnerabilities, which can be exploited to violate one or more of the software's security properties. According to CERT, most successful attacks result from targeting and exploiting known, non-patched software vulnerabilities and insecure software configurations, many of which are introduced during design and code.

The problems of non-secure software can be summed up as follows:

Software development is not yet a science or a rigorous discipline, and the development process is not controlled to minimize the vulnerabilities that attackers exploit. Today, as with cancer, vulnerable software can be invaded and modified to cause damage to previously healthy software, and infected software can replicate itself and be carried across networks to cause damage in other systems. Like cancer, these damaging processes may be invisible to the lay person even though experts recognize that their threat is growing. And as in cancer,

both preventive actions and research are critical, the former to minimize damage today and the latter to establish a foundation of knowledge and capabilities that will assist the cyber security professionals of tomorrow reduce risk and minimize damage for the long term.

The security of software is threatened at various points throughout its life cycle, both by inadvertent and intentional choices and actions taken by "insiders"–individuals closely affiliated with the organization that is producing, deploying, operating, or maintaining the software, and thus trusted by that organization–and by "outsiders" who have no affiliation with the organization. The software's security can be threatened

- **during its development:** A developer may corrupt the software–intentionally or unintentionally–in ways that will compromise the software's dependability and trustworthiness when it is operational.
- **during its deployment (distribution and installation):** If those responsible for distributing the software fail to tamperproof the software before shipping or uploading, or transmit it over easily intercepted communications channels, they leave the software vulnerable to intentional or unintentional corruption. Similarly, if the software's installer fails to "lock down" the host platform, or configures the software insecurely, the software is left vulnerable to access by attackers.
- **during its operation:** Once COTS (commercial off-the-shelf) and open source software (OSS) has gone operational, vulnerabilities may be discovered and publicized; unless security patches and updates are applied and newer supported versions (from which the root causes of vulnerabilities have been eliminated) are adopted, such software will become increasingly vulnerable. Non-commercial software and open source software (OSS) may also be vulnerable, especially as it may manifest untrustworthy behaviour over time due to changes in its environment that stress the software in ways that were not anticipated and simulated during its testing. Any software system that runs on a network-connected platform has its vulnerabilities exposed during its operation. The level of exposure will vary depending on whether the network is public or private, Internet-connected or not, and whether the software's environment has been configured to minimize its exposure. But even in highly controlled networks and "locked down" environments, the software may be threatened by malicious insiders (users, administrators, etc.).

- **during its maintenance:** If those responsible for addressing discovered vulnerabilities in released software fail to issue patches or updates in a timely manner, or fail to seek out and eliminate the root causes of the vulnerabilities to prevent their perpetuation in future releases of the software, the software will become increasingly vulnerable to threats over time. Also, the software's maintainer may prove to be a malicious insider, and may embed malicious code, exploitable flaws, etc., in updated versions of the code.

Both research and real-world experience indicate that correcting weaknesses and vulnerabilities as early as possible in the software's life cycle is far more cost-effective over the lifetime of the software than developing and releasing frequent security patches for deployed software.

Notes

CERT (Computer Emergency Rresponse Team) – група комп'ютерної «швидкої допомоги» (*організація, що слідкує за загрозами безпеці мережевих комп'ютерів, зокрема – в інтернеті*)

lay – непрофесійний

Assignments

1. Read and translate text 16.
2. Ask questions on the text.
3. Compose a summary of the text.

Text 17. Vulnerability Research

Vulnerability researchers tend to pride themselves on two things: finding vulnerabilities in code that really matters (especially if it has been heavily audited), and finding obscure and subtle vulnerabilities that are sufficiently complicated that it's hard to describe them in words without a few Visio diagrams. (Pro-tip: never open Visio documents from vulnerability researchers. Buy a whiteboard.)

Application security consultants tend to pride themselves on related things. You get dropped in the middle of a million line application with a team of potentially hostile developers and you do your best to quickly assemble a mental model of how a business works, what the software does, and how it can be attacked. You never have even remotely close to enough time, so you have to make your best Herculean effort to try and find all of the problems you can across all layers of abstraction. If

you're good at it, this necessarily causes you to discard your preconceived notions and find strategies that work. You have to be both deep and broad, and if you miss something that shows up a month later, well, you better hope your salesmen managed to get across the nuance of how it is impossible to prove a negative.

Any good application security consultant or vulnerability researcher would better be able to find both implementation and design flaws, and everything in between.

To put it another way, if you need an application security consultant and had the choice between hiring someone that could find security vulnerabilities by reverse engineering or hiring someone who was fully versed in design and policy review, you would better hire both of them.

Assignments

1. Read and translate text 17.
2. Ask questions on the text.
3. Speak on vulnerability experts.

Text 18. Software Security Assurance

The main objective of software assurance is to ensure that the processes, procedures, and products used to produce and sustain the software conform to all requirements and standards specified to govern those processes, procedures, and products. Software security and secure software are often discussed in the context of software assurance. Software assurance in its broader sense refers to the assurance of any required property of software. For software practitioners at the National Aeronautics and Space Administration (NASA), software assurance refers to the assurance of safety as a property of software. Similarly, in other communities, software assurance may refer to assurance of reliability or quality. In the context of this article, software assurance is concerned with assuring the *security* of software.

An increasingly agreed-upon approach for assuring the security of software is the software security assurance case, which is intended to provide justifiable confidence that the software under consideration (1) is free of vulnerabilities; (2) functions in the "intended manner," and this "intended manner" does not compromise the security or any other required properties of the software, its environment, or the information it handles; and (3) can be trusted to continue operating dependably un-

der all anticipated circumstances, including anomalous and hostile environmental and utilization circumstances—which means that those who build the software need to anticipate such circumstances and design and implement the software to be able to handle them gracefully. Such circumstances include

- the presence of unintentional faults in the software and its environment,
- the exposure of the operational software to accidental events that threaten its security,
- the exposure of the software to intentional choices or actions that threaten its security during its development, deployment, operation, or sustainment.

Software is more likely to be assurably secure when security is a key factor in the following aspects of its development and deployment:

- **development principles and practices:** The practices used to develop the software and the principles that governed its development are expressly intended to encourage and support the consideration and evaluation of security in every phase of the software's development life cycle.
- **development tools:** The programming language(s), libraries, and development tools used to design and implement the software are evaluated and selected for their ability to avoid security vulnerabilities and to support secure development practices and principles.
- **testing practices and tools:** The software is expressly tested to verify its security, using tools that assist in such testing.
- **acquired components:** Commercial off-the-shelf (COTS) and OSS (open source software) components are evaluated to determine whether they contain vulnerabilities, and if so whether the vulnerabilities can be remediated through integration to minimize the risk they pose to the software system.
- **deployment configuration:** The installation configuration of the software minimizes the exposure of any residual vulnerabilities it contains.
- **execution environment:** Protections are provided by the execution environment that can be leveraged to protect the higher level software that operates in that environment.
- **practitioner knowledge:** The software's analysts, designers, developers, testers, and maintainers are provided with the necessary informa-

tion (e.g., through training and education) to give them sufficient security awareness and knowledge to understand, appreciate, and effectively adopt the principles and practices that will enable them to produce secure software.

Assignments

1. Read and translate text 18.
2. Ask key questions on the text.
3. Make up a plan of the text.

Text 19. Secure Software Development Principles

The following principles should guide the development of secure software, including all decisions made in producing the artifacts at every phase of the software life cycle.

Minimize the number of high-consequence targets. The software should contain as few high-consequence targets (critical and trusted components) as possible. High-consequence targets are those that represent the greatest potential loss if the software is compromised and therefore require the most protection from attack. Critical and trusted components are high-consequence because of the magnitude of impact if they are compromised. *(This principle contributes to trustworthiness and, by its implied contribution to smallness and simplicity, also to dependability.)*

Don't expose vulnerable and high-consequence components. The critical and trusted components the software contains should not be exposed to attack. In addition, known vulnerable components should also be protected from exposure because they can be compromised with little attacker expertise or expenditure of effort and resources. *(This principle contributes to trustworthiness.)*

Deny attackers the means to compromise. The software should not provide the attacker with the means by which to compromise it. Such "means" include exploitable weaknesses and vulnerabilities, dormant code, backdoors, etc. Also, provide the ability to minimize damage, recover, and reconstitute the software as quickly as possible following a compromising (or potentially compromising) event to prevent greater compromise. In practical terms, this will require building in the means to monitor, record, and react to how the software behaves and what inputs it receives. *(This principle contributes to dependability, trustworthiness, and resilience.)*

Always assume "the impossible" will happen. Events that seem to be impossible rarely are. They are often based on an expectation that something in a particular environment is highly unlikely to exist or to happen. If the environment changes or the software is installed in a new environment, those events may become quite likely. The use cases and scenarios defined for the software should take the broadest possible view of what is possible. The software should be designed to guard against both likely and *un*likely events.

Developers should make an effort to recognize assumptions they are not initially conscious of having made and should determine the extent to which the "impossibilities" associated with those assumptions can be handled by the software. Specifically, developers should always assume that their software will be attacked, regardless of what environment it may operate in. This includes acknowledgement that environment-level security measures such as access controls and firewalls, being composed mainly of software themselves (and thus equally likely to harbor vulnerabilities and weaknesses), can and will be breached at some point, and so cannot be relied on as the sole means of protecting software from attack.

Developers who recognize the constant potential for their software to be attacked will be motivated to program defensively, so that software will operate dependably not only under "normal" conditions but under anomalous and hostile conditions as well. Related to this principle are two additional principles about developer assumptions.

1. Never make blind assumptions. Validate every assumption made by the software or about the software *before* acting on that assumption.

2. Security software is not the same as secure software. Just because software performs information security-related functions does not mean the software itself is secure. Software that performs security functions is just as likely to contain flaws and bugs as other software. However, because security functions are high-consequence, the compromise or intentional failure of such software has a significantly higher potential impact than the compromise or failure of other software.

Assignments

1. Read and translate text 19.
2. Ask questions on the text.
3. Speak on the principles of secure software development.

Text 20. What a Software Practitioner Needs to Know

The main characteristics that discriminate the developer, tester, integrator, and sustainer of secure software from those of non-secure software are awareness, intention, and caution. A software professional who cares about security and acts on that awareness will recognize that software vulnerabilities and weaknesses can originate at any point in the software's conception or implementation, from inadequate requirements, to poor design and implementation choices, to inadvertent coding errors or configuration mistakes.

The security-aware software professional knows that the only way these problems can be avoided is through well-informed and intentional effort: requirements analysts must understand how to translate the need for software to be secure into actionable requirements, designers must recognize choices that conflict with secure design principles, and programmers must follow secure coding practices and be cautious about avoiding coding errors and finding and removing the bugs they were unable to avoid. Software integrators must recognize and strive to reduce the security risk associated with vulnerable components (whether custom-built, COTS, or open source), and must understand the ways in which those modules and components can be integrated to minimize the exposure of any vulnerabilities that cannot be eliminated.

The main reason for adding security practices throughout the software development life cycle (SDLC) is to establish a software life cycle process that codifies both caution and intention.

Creating a secure development community using collaboration technologies and a well-integrated development environment promotes a continuous process of improvement and a focus on secure development life cycle principles and practices that will result in the ongoing production of more dependable, trustworthy, survivable software systems.

Assignments

1. Read and translate text 20.
2. Ask key questions on the text.
3. Write a summary of the text.

Text 21. Integrating Security into the Software Life Cycle

"Security enhancement" of the SDLC process mainly involves the adaptation or augmentation of existing SDLC activities, practices, and check-

points, and in a few instances, it may also entail the addition of new activities, practices, or checkpoints. In a very few instances, it may also require the elimination or wholesale replacement of certain activities or practices that are known to obstruct the ability to produce secure software.

The key elements of a secure software life cycle process are:

- security criteria in all software life cycle checkpoints (both at the entry of a life cycle phase and at its exit);
- adherence to secure software principles and practices;
- adequate requirements, architecture, and design;
- secure coding practices;
- secure software integration/assembly practices;
- security testing practices that focus on verifying the dependability, trustworthiness, and sustainability of the software being tested;
- secure distribution and deployment practices and mechanisms;
- secure maintenance practices;
- supportive tools;
- secure software configuration management systems and processes;
- security-knowledgeable software professionals;
- security-aware project management;
- upper management commitment to production of secure software.

Organizations can insert secure development practices into their software life cycle process by adopting a codified secure software development methodology. This would help them in making progress toward achieving their goals.

Assignments

1. Read and translate text 21.
2. Ask key questions on the text.
3. Speak on the key elements of a secure software life cycle process.

Text 22. Software Security Engineering

Software is ubiquitous. Many of the products, services, and processes organizations use and offer are highly dependent on software to handle the sensitive and high-value data on which people's privacy, livelihoods, and very lives depend. National security – and by extension citizens' personal safety – relies on increasingly complex, interconnected, software-intensive information systems – systems that in many cases use the Internet or Internet-exposed private networks as their means for communication and transporting data.

Dependence on information technology makes software security a key element of business continuity, disaster recovery, incident response, and national security. Software vulnerabilities can jeopardize intellectual property, consumer trust, business operations and services, and a broad spectrum of critical applications and infrastructures, including everything from process control systems to commercial application products.

The integrity of critical digital assets (systems, networks, applications, and information) depends on the reliability and security of the software that enables and controls those assets. However, business leaders and informed consumers have growing concerns about the scarcity of practitioners with requisite competencies to address software security. They have concerns about suppliers' capabilities to build and deliver secure software that they can use with confidence and without fear of compromise. Application software is the primary gateway to sensitive information.

The absence of security discipline in today's software development practices often produces software with exploitable weaknesses. Security-enhanced processes and practices – and the skilled people to manage them and perform them – are required to build software that can be trusted to operate more securely than software being used today.

There is an economic counter-argument, or at least the perception of one. Some business leaders and project managers believe that developing secure software slows the process and adds to the cost while not offering any apparent advantage. In many cases, when the decision reduces to "ship now" or "be secure and ship later," "ship now" is almost always the choice made by those who control the money but have no idea of the risks. Information to combat this argument, including how software security can potentially reduce cost and schedule, is becoming available based on earlier work in software quality and the benefits of detecting software defects early in the life cycle along with documented experiences such as Microsoft's Security Development Lifecycle.

Software security engineering is using practices, processes, tools, and techniques that enable you to address security issues in every phase of the software development life cycle (SDLC). Software that is developed with security in mind is typically more resistant to both intentional attack and unintentional failures. One view of secure software is software that is engineered "so that it continues to function correctly under malicious attack" and is able to recognize, resist, tolerate, and recover from events that intentionally threaten its dependability. Broader views

that can overlap with software security (for example, software safety, reliability, and fault tolerance) include proper functioning in the face of unintentional failures or accidents and inadvertent misuse and abuse, as well as reducing software defects and weaknesses to the greatest extent possible regardless of their cause.

The goal of software security engineering is to build better, defect-free software. Software-intensive systems that are constructed using more securely developed software are better able to:

- continue operating correctly in the presence of most attacks by either *resisting* the exploitation of weaknesses in the software by attackers or *tolerating* the failures that result from such exploits;
- limit the damage resulting from any failures caused by attack-triggered faults that the software was unable to resist or tolerate and recover as quickly as possible from those failures.

Assignments

1. Read and translate text 22.
2. Ask questions on the text.
3. Give the gist of the text.

Text 23. Software Security Practices

No single practice offers a universal silver bullet for software security. The objective is to increase the security and dependability of the software produced by these practices, both during its development and its operation.

Software developed and assembled using software security practices should contain significantly fewer exploitable weaknesses. Such software can then be relied on to more capably recognize, resist or tolerate, and recover from attacks and thus function more securely in an operational environment.

The five key rules of Software Security Engineering are as follows:

1. Software security is about more than eliminating vulnerabilities and conducting penetration tests. Project managers need to take a systematic approach to incorporate the sound software security practices into their development processes. Examples include security requirements elicitation, attack pattern and misuse/abuse case definition, architectural risk analysis, secure coding and code analysis, and risk-based security testing.

2. Network security mechanisms and IT infrastructure security services do not sufficiently protect application software from security risks.

3. Software security initiatives should follow a risk management approach to identify priorities and what is good enough, understanding that software security risks will change throughout the development lifecycle. Risk management reviews and actions are conducted during each phase of the SDLC.

4. Developing secure software depends on understanding the operational context in which it will be used. This context includes conducting end-to-end analysis of cross-system work processes, working to contain and recover from failures using lessons learned from business continuity, and exploring failure analysis and mitigation to deal with system complexity.

5. Project managers and software engineers need to learn to think like an attacker in order to address the range of things that software should not do and how software can better resist, tolerate, and recover when under attack. The use of attack patterns and misuse/abuse cases throughout the SDLC encourages this perspective.

Assignments

1. Read and translate text 23.
2. Ask questions on the text.
3. Speak on software security practices.

Text 24. Data Erasure and Recovery

Data erasure is a method of software-based overwriting that completely destroys all electronic data residing on a hard disk drive or other digital media. Permanent data erasure goes beyond basic file deletion commands, which only remove direct pointers to data disk sectors and make data recovery possible with common software tools. Unlike degaussing and physical destruction, which render the disk unusable, data erasure removes all information while leaving the disk operable, preserving assets and the environment.

Software-based overwriting uses a software application to write patterns of meaningless data onto each of a hard drive's sectors. There are key differentiators between data erasure and other overwriting methods, which can leave data intact and raise the risk of data breach or spill, identity theft and failure to achieve regulatory compliance. Data erasure also provides multiple overwrites so that it supports recognized government and industry standards. It provides verification of data removal, which is necessary for meeting certain standards.

To protect data on lost or stolen media, some data erasure applications remotely destroy data if the password is incorrectly entered. Data erasure tools can also target specific data on a disk for routine erasure, providing a hacking protection method that is less time-consuming than encryption.

Data recovery is the process of salvaging data from damaged, failed, corrupted, or inaccessible secondary storage media when it cannot be accessed normally. Often the data are being salvaged from storage media such as hard disk drives, storage tapes, CDs, DVDs, RAID, and other electronics. Recovery may be required due to physical damage to the storage device or logical damage to the file system that prevents it from being mounted by the host operating system (OS).

The most common "data recovery" issue involves an operating system failure (typically on a single-disk, single-partition, single-OS system), where the goal is to simply copy all wanted files to another disk. This can be easily accomplished with a Live CD, most of which provides a means to 1) mount the system drive, 2) mount and backup disk or media drives, and 3) move the files from the system to the backup with a file manager or optical disc authoring software. Further, such cases can be mitigated by disk partitioning and consistently moving valuable data files to a different partition from the replaceable OS system files.

The second type involves a disk-level failure such as a compromised file system, disk partition, or a hard disk failure – in each of which the data cannot be easily read. Depending on the case, solutions involve repairing the file system, partition table or MBR, or hard disk recovery techniques ranging from software-based recovery of corrupted data to hardware replacement on a physically damaged disk. These last two typically indicate the permanent failure of the disk, thus "recovery" means sufficient repair for a one-time recovery of files.

A third type involves the process of retrieving files that have been "deleted" from a storage media, since the files are usually not erased in any way but are merely deleted from the directory listings.

Although there is some confusion as to the term, the term "data recovery" may be used to refer to such cases in the context of forensic purposes or spying.

Notes

RAID (Redundant Array of Inexpensive (Independent) Disks) дисковий масив (*матриця*) – система зовнішньої пам'яті, масив недорогих (*або незалежних*) дисків з надлишковістю

MBR (master boot record) – 1) головний завантажувальний запис; 2) memory buffer register – регістр буфера пам'яті, регистр MBR

Assignments

1. Read and translate text 24.
2. Ask questions on the text.
3. Write a summary of the text.

LITERATURE

1. *Пройдаков Е. М.* Англо-український тлумачний словний з обчислювальної техніки, Інтернету і програмування / Е. М. Пройдаков, Л. А. Теплицький. – Вид. 1. – К. : Видавн. дім «Софт-Прес», 2005. – 552 с.

2. *Смирнова Т. В.* English for Computer Science Students / Т. В. Смирнова, М. В. Юдельсон. – М. : Флинта: Наука, 2002. – 128 с.

3. *Радовель В. А.* Английский язык. Основы компьютерной грамотности / В. А. Радовель. – Ростов-на-Дону : Феникс, 2006. – 224 с.

4. *English*. Types of Modern Computers : Methodological Guide / O. Ye. Bugaiov, H. V. Babiy, Ya. V. Absaliamova. – Kyiv : National Aviation University, 2003. – 68 p.

5. *Fundamentals* of Information Technology / Ed. By G. G. Wilkinson, A. R. Winterflood. – Chichester : John Wiley and Sons, 1987. – 363 p.

6. *Oxford* English for Information Technology / E. H. Glendinning, J. McEwan. – Oxford Press, 2003. – 222 p.

7. *Professional* English in Use for Computers and the Internet / S. R. Esteras, E. M. Fabre. – Cambridge University Press, 2007. – 118 p.

8. *Sidorov M. O*. Software Engineering. – Lecture Course. – Kyiv : NAU, 2007. – 139 p.

9. *Sidorov N. A*. Basics of Programming Languages. – Lecture Course. – Kyiv : NAU, 2003. – 130 p.

10. http://www.wikipedia.com

11. http://www.britannica.com

12. http://www.excelsoftware.com

13. http://cc2e.com

CONTENTS

Навчальне видання

АКМАЛДІНОВА Олександра Миколаївна
БУГАЙОВ Олександр Євгенович
ТЕРЕМІНКО Лариса Григорівна
ГУРСЬКА Олена Олександрівна
МИСЛИВА Тетяна Анатоліївна
МУРКІНА Наталія Іванівна

PROFESSIONAL ENGLISH FUNDAMENTALS OF SOFTWARE ENGINEERING

Навчальний посібник

Редактор *З. О. Остап'юк*
Технічний редактор *А. І. Лавринович*
Художник обкладинки *Л. В. Карпук*
Комп'ютерна верстка *Н. В. Чорної*

Підп. до друку 18.09.2015. Формат 60x84/16. Папір офс.
Офс. друк. Ум. друк. арк. 17,44. Обл.-вид. арк. 18,75.
Тираж 100 пр. Замовлення № 165-1.

Видавець і виготівник
Національний авіаційний університет
03680. Київ-58, проспект Космонавта Комарова, 1.

Свідоцтво про внесення до Державного реєстру ДК № 977 від 05.07.2002