Питання до ККР з об'єктно-орієнтованого програмування (ООП)

Напрям: 6.050102 "Комп'ютерна інженерія"

Спеціальність: 6.05010202 "Системне програмування"

Варіант 1.

1. Визначте основні принципи ООП.
2. Проаналізуйте використання таких засобів форматного введення-виведення у C++, як: прапори формату, функції setf() і unsetf().
3. У наведеному коді перевантажте оператор "+" для додавання двох об'єктів класу *coord*:

```
class coord {
   int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
  void main() { //… оголошення об'єктів
  // … додавання об'єктів - виклик
              //оператора-функції
  }
```

Варіант 2.

1. Визначте принцип ООП — інкапсуляцію.
2. Проаналізуйте поняття віртуальної функції як одного із видів реалізації динамічного поліморфізму та визначте поняття поліморфного класу.
3. У наведеному коді перевантажте оператор "-" для віднімання двох об'єктів класу coord:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
  // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об'єктів
// … віднімання об'єктів - виклик
              //оператора-функції
}
```

Варіант 3.

1. Визначте поняття класу і об’єкту в ООП.
2. Проаналізуйте поняття чисто віртуальної функції як одного із видів реалізації динамічного поліморфізму та визначте поняття абстрактного класу.
3. У наведеному коді перевантажте оператор “*” для множення двох об’єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
  // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … множення об’єктів - виклик оператора-функції
}
```

Варіант 4.

1. Проаналізуйте відмінності відкритих (public) і закритих (private) членів класу в ООП.
2. Проаналізуйте процес оброблення виняткових ситуацій у мові C++, їх генерацію і перехоплення.
3. У наведеному коді перевантажте оператор “/” для ділення двох об’єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
  // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … ділення об’єктів - виклик оператора-функції
}
```

Варіант 5.

1. Визначте загальні форми (синтаксис) описання класу та його функцій-членів.
2. Визначте поняття родового класу. Розкрийте особливості оголошення об'єктів такого класу.
3. У наведеному коді перевантажте оператор "==" для двох об'єктів класу *coord*:

```
class coord {
    int x,y;
  public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
    void get_xy(int &i, int &j) { i = x; j = y; }
   // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об'єктів
// … виклик оператора-функції
}
```

Варіант 6.

1. Визначте поняття конструктора класу. Проаналізуйте, коли і як викликаються конструктори?
2. Визначте особливості використання стандартних маніпуляторів у C++.
3. У наведеному коді перевантажте оператор "++" для об'єкта класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об'єктів
// … інкрементація об'єкта - виклик
                //оператора-функції
}
```

Варіант 7.

1. Поясніть, що відбувається з відкритими (public), захищеними (protected) та закритими (private) членами базового класу, якщо базовий клас успадковується як відкритий (public) похідним?
2. Проаналізуйте особливості перевантаження бінарних операторів.
3. Використовуючи принципи потокової системи введення-виведення інформації у C++, напишіть програму в якій: 1) створіть вихідний файл для запису; 2) запишіть до файлу число 100 в десятковій та шістнадцятковій системах числення; 3) закрийте файл; 4) знову відкрийте, але вже як вхідний файл для зчитування; 5) зчитайте інформацію у змінні відповідних типів даних і виведіть її на екран.

Варіант 8.

1. Визначте поняття деструктора класу. Проаналізуйте, коли і як викликаються деструктори?
2. Проаналізуйте особливості використання аргументів за замовчуванням.
3. У наведеному коді перевантажте оператор “&&” для двох об’єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
  // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … виклик оператора-функції
}
```

Варіант 9.

1. Проаналізуйте використання конструкторів без та з параметрами.
2. Визначте загальні форми (синтаксис) і функції таких операторів мови C++, як: try, catch і throw.
3. У наведеному коді перевантажте оператор “--” для об’єкта класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … декрементація об’єкта - виклик
                //оператора-функції
}
```

Варіант 10.

1. Визначте, як визначаються і викликаються функції-члени класу.
2. Проаналізуйте особливості множинного успадкування у C++.
3. У наведеному коді перевантажте оператор “+” як дружню функцію для додавання цілого числа і об’єкта класу *coord* (int + coord):

```
class coord {
    int x,y;
  public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
    void get_xy(int &i, int &j) { i = x; j = y; }
   // ... оголошення оператора-функції
};
//... визначення оператора-функції для int + coord
  void main() { //… оголошення об’єктів
  // … додавання числа і об’єкта - виклик
                     //оператора-функції
}
```

Варіант 11.

1. Визначте особливості використання вбудованих функцій, їх переваги та недоліки.
2. Визначте особливості виклику конструкторів при множинному успадкуванні.
3. У наведеному коді перевантажте оператор вилучення “>>” для об’єкту класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції вилучення
};
//... визначення оператора-функції вилучення
void main() { //… оголошення об’єктів
// … виклик оператора-функції вилучення
}
```

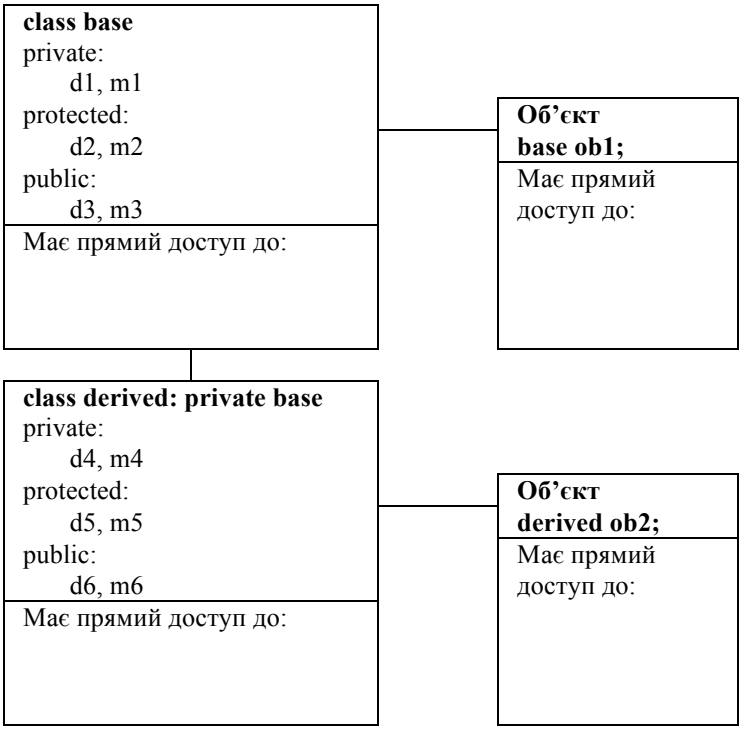
Варіант 12.

1. Обгрунтуйте особливості використання дружніх функцій.
2. Визначте поняття родової функції та наведіть типову форму (синтаксис) її визначення.
3. У наведеному коді перевантажте оператор вставки “<<” для об’єкту класу coord:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції вставки
};
//... визначення оператора-функції вилучення
void main() { //… оголошення об’єктів
// … виклик оператора-функції вставки
}
```

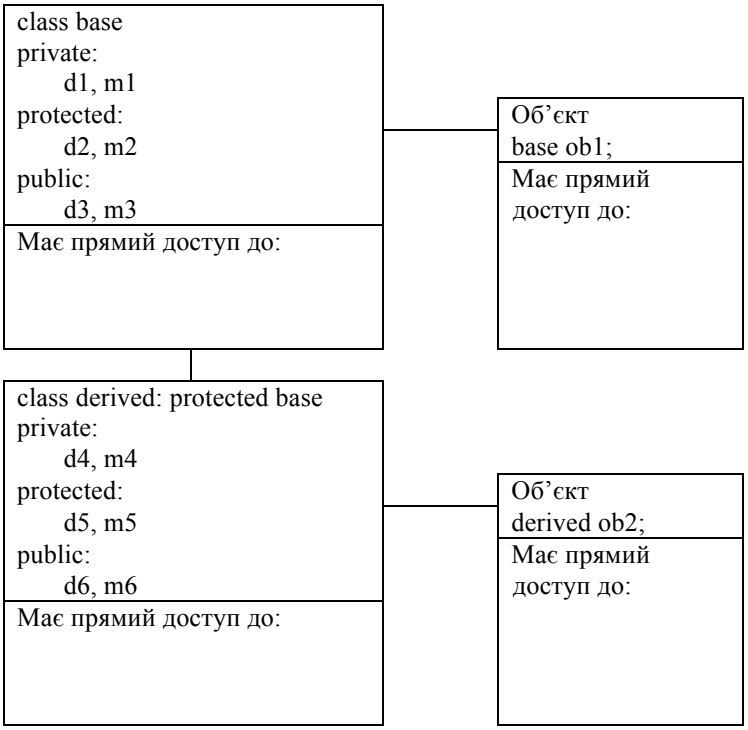
Варіант 13.

1. Визначте принцип ООП — поліморфізм.
2. Проаналізуйте використання таких засобів форматного введення-виведення у C++, як: функції width(), precision(), fill().
3. Заповніть пропущені ідентифікатори, вказавши у коментарях пояснення:



Варіант 14.

1. Проаналізуйте особливості вказівників на базові та похідні класи.
2. Визначте загальні форми (синтаксис) оператора-функції члена класу та дружньої функції-оператора.
3. Заповніть пропущені ідентифікатори, вказавши у коментарях пояснення:

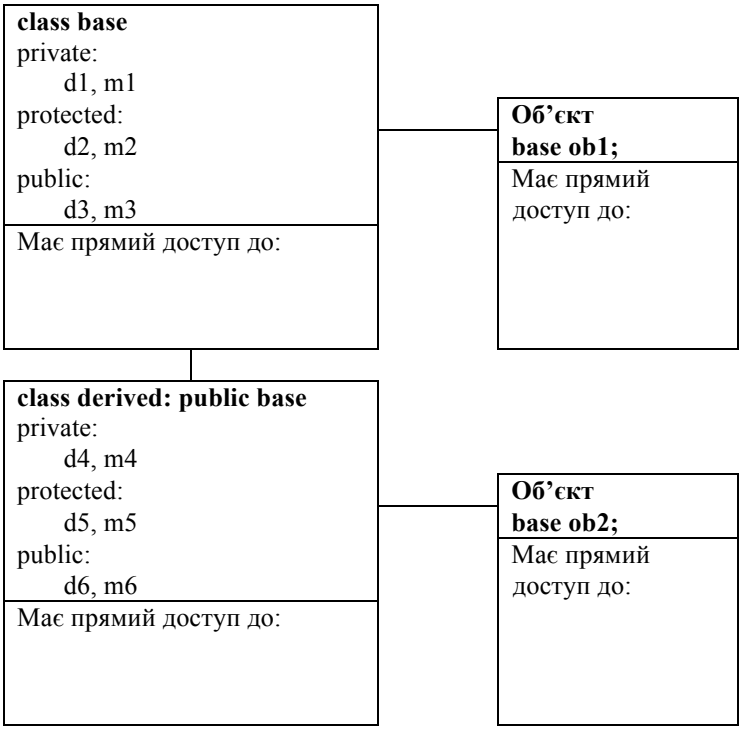


Варіант 15.

1. Визначте поняття класу і об'єкту в ООП.
2. Проаналізуйте особливості перевантаження бінарних операторів.
3. Напишіть родову функцію з двома родовими типами даних, яка виводить значення двох своїх аргументів на екран. Продемонструйте виклик створеної функції у функції *main()*.

Варіант 16.

1. Визначте поняття деструктора класу. Проаналізуйте, коли і як викликаються деструктори?
2. Проаналізуйте особливості перевантаження унарних операторів.
3. Заповніть пропущені ідентифікатори, вказавши у коментарях пояснення:



Варіант 17.

1. Визначте принцип ООП — інкапсуляцію.
2. Проаналізуйте особливості перевантаження операторів відношення та логічних операторів.
3. Напишіть у поданому нижче коді віртуальну функцію *func()*, яка: 1) у базовому класі *base* виводить значення змінної *i* на екран; 2) у похідному класі *derived1* виводить значення квадрату змінної *i* базового класу; 3) у похідному класі *derived2* не перевизначається. Напишіть результат виконання програми:

```
#include<iostream.h>
class base {
  public:
    int i;
    base(int x) {i=x;}
    // додайте визначення віртуальної функції
};
class derived1 : public base {
  public:
    derived1(int x) : base(x) { }
    // додайте визначення віртуальної функції
};
class derived2 : public base {
  public:
    derived2(int x) : base(x) { }
// у derived2 функція func() не перевизначається
};
void main(){
  base *p;
  base ob(10);
  derived1 d_ob1(10);
  derived2 d_ob2(10);
  p=&ob;
  p->func( );
  p=&d_ob1;
  p->func( );
  p = &d_ob2;
  p->func( );
}
```

Варіант 18.

1. Обгрунтуйте використання захищених (protected) членів класу під час успадкування.
2. Проаналізуйте відмінності між звичайними віртуальними та чисто віртуальними функціями, між поліморфними і абстрактними класами.
3. У наведеному коді перевантажте оператор “+” для додавання двох об’єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
  // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … додавання об’єктів - виклик
           //оператора-функції
}
```

Варіант 19.

1. Проаналізуйте особливості доступу до елементів базового класу під час успадкування.
2. Проаналізуйте переваги використання контейнерних класів.
3. У наведеному коді перевантажте оператор “-” для віднімання двох об’єктів класу coord:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … віднімання об’єктів - виклик
              //оператора-функції
}
```

Варіант 20.

1. Проаналізуйте відмінності між віртуальними та перевантаженими функціями.
2. Визначте особливості оголошення і використання власних маніпуляторів користувача у C++.
3. У наведеному коді перевантажте оператор “*” для множення двох об’єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … множення об’єктів - виклик оператора-функції
}
```

Варіант 21.

1. Визначте принцип ООП — успадкування.
2. Визначте особливості оголошення і використання власних функцій вставки у C++.
3. У наведеному коді перевантажте оператор “&&” для двох об’єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … виклик оператора-функції
}
```

Варіант 22.

1. Поясніть, що відбувається з відкритими (public), захищеними (protected) та закритими (private) членами базового класу, якщо базовий клас успадковується як відкритий (public) похідним?
2. Визначте особливості оголошення і використання власних функцій вилучення у C++.
3. У наведеному коді перевантажте оператор "/" для ділення двох об'єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об'єктів
// … ділення об'єктів - виклик оператора-функції
}
```

Варіант 23.

1. Поясніть, що відбувається з відкритими (public), захищеними (protected) та закритими (private) базового класу, якщо базовий клас успадковується як закритий (private) похідним?
2. Проаналізуйте особливості файлового введення-виведення у C++.
3. У наведеному коді перевантажте оператор "+" як дружню функцію для додавання об'єкта класу *coord* і цілого числа (coord + int):

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
  // ... оголошення оператора-функції
};
//... визначення оператора-функції для coord + int
void main() { //… оголошення об'єктів
// … додавання об'єкта і числа - виклик
                    //оператора-функції
}
```

Варіант 24.

1. Поясніть, що відбувається з відкритими (public), захищеними (protected) та закритими (private) базового класу, якщо базовий клас успадковується як захищений (protected) похідним?
2. Проаналізуйте відмінності між родовими та перевантаженими функціями?
3. Напишіть програму, яка виводить на екран таку інформацію у відповідному форматі:

				П	**р**	**и**	**в**	**і**	**т**
*	*	*	*	**П**	**р**	**и**	**в**	**і**	**т**
П	**р**	**и**	**в**	**і**	**т**	*	*	*	*
1	**2**	**3**	**.**	**2**	**3**	**4**	**5**	**6**	**7**
1	**2**	**3**	**.**	**2**	**3**	**4**	*	*	*

Варіант 25.

1. Визначте основні принципи ООП.
2. Обгрунтуйте і визначте особливості родових функцій.
3. Напишіть у поданому нижче коді віртуальну функцію *func()*, яка: 1) у базовому класі *base* виводить значення змінної *i* на екран; 2) у похідному класі *derived1* виводить значення квадрату змінної *i* базового класу; 3) у похідному класі *derived2* виводить подвійне значення змінної *i*. Напишіть результат виконання програми:

```
#include <iostream.h>
class base {
  public:
    int i;
    base(int x) {i=x;}
    // додайте визначення віртуальної функції
};
class derived1 : public base {
  public:
    derived1(int x) : base(x) { }
    // додайте визначення віртуальної функції
};
class derived2 : public base {
  public:
    derived2(int x) : base(x) { }
    // додайте визначення віртуальної функції
};
void main(){
  base *p;
  base ob(10);
  derived1 d_ob1(10);
  derived2 d_ob2(10);
  p=&ob;
  p->func();
  p=&d_ob1;
  p->func();
  p=&d_ob2;
  p->func();
}
```

Варіант 26.

1. Проаналізуйте один з видів реалізації статичного поліморфізму — перевантаження функцій.
2. Проаналізуйте особливості застосування оператора catch(…) мови C++.
3. Напишіть програму у якій: 1) відкрийте вхідний текстовий файл для зчитування; 2) створіть вихідний текстовий файл для запису; 3) скиньте прапор *skipws* вхідного файлового потоку; 4) скопіюйте текстовий файл зчитування у файл запису і при цьому перетворіть пробіли на символи "|"; використайте функцію *eof()* для контролю кінця вхідного файлу зчитування.

Варіант 27.

1. Проаналізуйте один з видів перевантаження функцій — перевантаження конструкторів.
2. Визначте особливості оголошення і використання власних функцій вставки у C++.
3. У наведеному коді перевантажте оператор "==" для двох об'єктів класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об'єктів
// … виклик оператора-функції
}
```

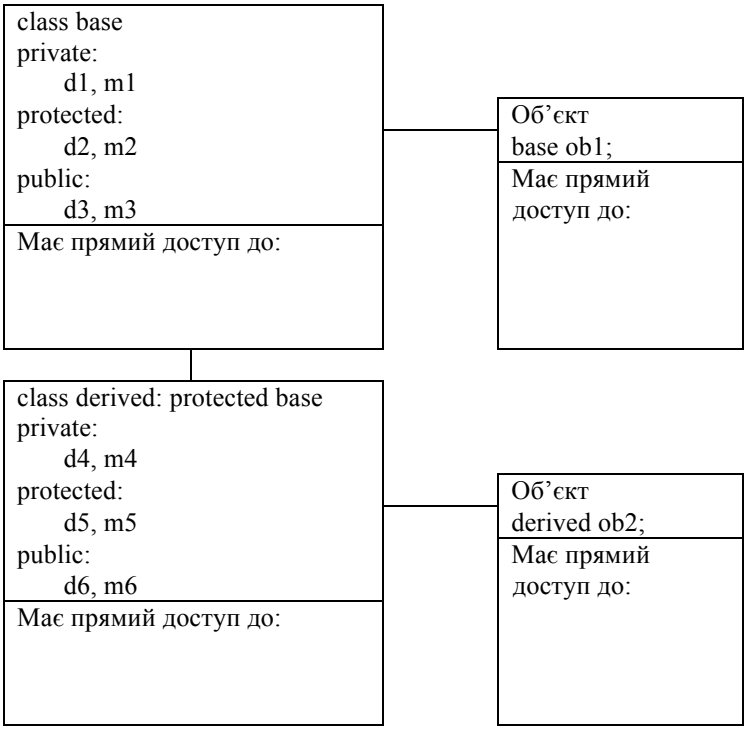
Варіант 28.

1. Визначте порядок виклику конструкторів і деструкторів при успадкуванні та особливості передачі аргументів конструкторам базового та похідного класів.
2. Проаналізуйте особливості системи введення-виведення мови С++.
3. У наведеному коді перевантажте оператор “++” для об’єкта класу *coord*:

```
class coord {
  int x,y;
 public:
  coord() { x = 0; y= 0; }
  coord(int i, int j) { x = i; y = j; }
  void get_xy(int &i, int &j) { i = x; j = y; }
 // ... оголошення оператора-функції
};
//... визначення оператора-функції
void main() { //… оголошення об’єктів
// … інкрементація об’єкта - виклик
                //оператора-функції
}
```

Варіант 29.

1. Проаналізуйте особливості перевантаження операторів як функцій-членів класу.
2. Визначте родову функцію кількох родових типів даних.
3. Заповніть пропущені ідентифікатори, вказавши у коментарях пояснення:



Варіант 30.

1. Проаналізуйте відмінності відкритих (public) і закритих (private) членів класу в ООП.
2. Проаналізуйте особливості перевантаження операторів як функцій, дружніх класу.
3. Напишіть родову функцію, яка міняє місцями значення двох своїх аргументів. Продемонструйте виклик створеної функції для типів *int*, *char* та *double* у функції *main()*.