Проектування і опрацювання програми з віртуальними функціями

Мета роботи — засвоєння поняття віртуальної функції; набуття практичних навичок їх оголошення та використання.

Основні завдання роботи

Розробити та реалізувати програмно код, у якому демонструється робота віртуальної функції. Для цього виконати такі завдання:

1. Розробити і реалізувати програмно клас “Число”.
2. Розробити ієрархію класів: *число* (базовий), *матриця* (похідний).
3. Розробити та реалізувати програмно віртуальну функцію, яка обчислює факторіал числа, заданого в базовому класі, і використовується похідним класом, що містить масив цілих чисел, факторіали яких слід підрахувати та вивести у таблицю.

Основні теоретичні відомості

1. Вказівники на базові класи

Основою практичного застосування віртуальних функцій та динамічного поліморфізму є вказівники на базові класи.

Важливою особливістю вказівників у C++ є те, що вказівник, оголошений як вказівник на базовий клас, також може використовуватися як вказівник на будь-який клас, похідний від цього базового. У такій ситуації такі оператори є правильними:

```
base *p; // вказівник базового класу
base base_ob; // об’єкт базового класу
derived derived_ob; // об’єкт похідного класу
p = &base_ob; // p вказує на об’єкт базового класу
p = &derived_ob; // p вказує на об’єкт похідного класу
```

Указівник на похідний клас неможливо використати для доступу до об’єктів базового класу.

2. Віртуальні функції

Віртуальна функція (virtual function) є функцією-членом класу, оголошується всередині базового класу та перевизначається в похідному класі. Для створення віртуальної функції перед оголошенням функції ставиться ключове слово

virtual. Якщо клас, що містить віртуальну функцію, успадковується, то в похідному класі віртуальна функція перевизначається.

Для перевизначення віртуальної функції в похідному класі ключове слово *virtual* не потрібне.

Віртуальна функція може викликатися так само, як і будь-яка інша функція-член класу. Проте більш цікавим використанням віртуальної функції є її виклик через вказівник, завдяки чому підтримується динамічний поліморфізм. Якщо вказівник базового класу вказує на об'єкт похідного класу, який містить віртуальну функцію і для якого віртуальна функція викликається через цей вказівник, то компілятор визначає, яку версію віртуальної функцію викликати на основі типу об'єкта, на який вказує вказівник. Цей процес є реалізацією принципу динамічного поліморфізму. Клас, що містить віртуальну функцію, називають **поліморфним класом** (polymorphic class).

Приклад. Використання віртуальної функції:

```
#include <iostream.h>
class base {
  public:    int i;
     base(int x) {i=x;}
     virtual void func()
     {    cout<<"Використання базової версії func():";

       cout<<i<<'\n'; }
};
class derived1 : public base {
    public:
       derived1(int x) : base(x) { }
       void func() {
        cout<<"Використання версії func() похідного класу
                                                    derived1:";
        cout<<i*i<<'\n'; }
};
class derived2 : public base {
   public:
      derived2(int x) : base(x) { }
      void func() {
       cout<<"Використання версії func() похідного
```

```
        класу derived2:";
        cout<<i+i<<'\n'; }
};
int main()
{
  base *p;
  base ob(10);
  derived1 d_ob1(10);
  derived2 d_ob2(10);
  p=&ob;
  p->func(); // Використання базової версії func()
  p=&d_ob1;
  p->func(); // Використання версії func() похідного
             // класу derived1
  p=&d_ob2;
  p->func(); // Використання версії func() похідного
           // класу derived2
  return 0;
}
```

Ця програма виводить на екран такий запис:

```
Використання базової версії func( ):10
Використання версії func( ) похідного класу derived1:100
Використання версії func( ) похідного класу derived2:20
```

Віртуальні функції мають ієрархічний порядок успадкування. Окрім цього, якщо віртуальна функція не перевизначається в похідному класі, то використовується її реалізація, задана в базовому класі.

3. Чисто віртуальні функції

Іноді, коли віртуальна функція оголошується в базовому класі, вона не виконує ніяких значущих дій. Це звичайна ситуація, оскільки часто базовий клас не визначає закінченого типу. Замість цього він просто вміщує базовий набір функцій-членів та змінних, для яких похідний клас задає все, чого не вистачає. Коли у віртуальній функції базового класу не виконується значуща дія, у реалізації будь-якого похідного класу ця функція має обов'язково перевизначатись. Для

реалізації цього положення C++ підтримує **чисто віртуальні функції** (pure virtual function).

Чисто віртуальні функції не визначаються в базовому класі. В нього включаються тільки прототипи цих функцій. Для чисто віртуальної функції використовується така загальна форма:

```
virtual тип_даних_що_повертається_функцією
                    ім'я_функції (список_параметрів) = 0;
```

Прирівняння функції до нуля — це повідомлення компілятору, що в базовому класі не існує тіла функції. Якщо функція задається як чисто віртуальна, то вона має перевизначатися в кожному похідному класі. Якщо цього немає, то під час компіляції виникає помилка. Таким чином, створення чисто віртуальних функцій гарантує, що похідні класи забезпечать їх перевизначення.

Якщо клас має хоча б одну чисто віртуальну функцію, то це буде **абстрактний клас** (abstract class). Оскільки в абстрактному класі є хоча б одна функція, яка не має тіла, технічно цей клас є неповним і жодного об'єкта цього класу створити не можна. Таким чином абстрактні класи можуть бути тільки успадкованими. Можна створювати вказівники абстрактного класу, завдяки яким досягається динамічний поліморфізм.

Порядок виконання роботи

1. Створити у програмі клас *Chislo*. В цьому класі задати закриту змінну у вигляді цілого додатного числа (типу *long*).

Визначити конструктор і деструктор цього класу.

Визначити віртуальну функцію *factorial()*, яка має обчислювати і повертати факторіал числа, яке передається йому як параметр.

2. Створити похідний клас *Matrix*, який успадковує *Chislo*. Цей клас в розділі *public* має містити одномірний масив 100 цілих додатних чисел, а успадковуване ціле число (вводиться з клавіатури) від базового класу має сенс розмірності цього масиву.

Визначити конструктор і деструктор цього класу.

Віртуальну функцію у похідному класі не перевизначати. Проаналізуйте отриманий результат.

3. Числа масиву вводяться з клавіатури користувачем. У програмі написати функцію обчислення факторіала кожного числа масиву тієї розмірності, яка зберігається у змінній базового класу.

4. Продемонструвати роботу створених функцій в основній частині програми. Використати вказівники на базові класи.

Контрольні завдання та запитання

1. Що таке віртуальна функція?
2. Які функції не можуть бути віртуальними?
3. Яка відмінність між віртуальною та перевантаженою функціями?
4. Як віртуальні функції допомагають реалізувати динамічний поліморфізм?
5. Що таке чисто віртуальна функція? Які її відмінності від звичайної віртуальної функції?
6. Що таке абстрактний клас? Що таке поліморфний клас?
7. Чи успадковується віртуальність?
8. У чому перевага динамічного поліморфізму?
9. Динамічний поліморфізм досягається за допомогою _____ функцій та вказівників на _____ класу. Впишіть пропущені слова.