

Лабораторна робота

№7

Тема: «Розробка програм з користувацькими класами. Робота з класами та об'єктами.»

Виконав: Устич Максим

Група: alk43

Київ 2025р.

Відповіді на контрольні питання.

1. Поясніть принцип ООП — успадкування. Наведіть приклади.

Успадкування — це один із головних принципів об'єктно-орієнтованого програмування.

Воно дозволяє створювати нові класи (похідні) на основі існуючих (базових).

Похідний клас успадковує всі властивості та методи базового класу, але може доповнювати їх або перевизначати.

Приклад:

```
#include <iostream>
using namespace std;

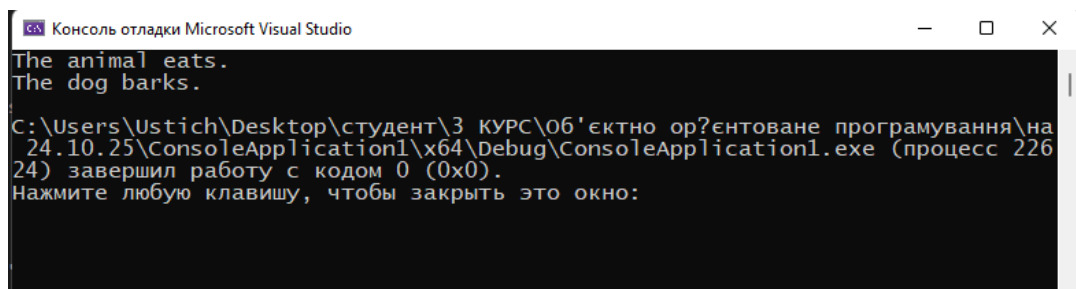
class Animal {
public:
    void eat() {
        cout << "The animal eats." << endl;
    }
};

class Dog : public Animal {
public:
    void bark() {
        cout << "The dog barks." << endl;
    }
};

int main() {
    Dog d;
    d.eat(); // успадкований метод
    d.bark(); // власний метод
    return 0;
}
```

Рис. 1. Код прикладу успадкування у Visual Studio.

Виконання програми:



```
Консоль отладки Microsoft Visual Studio
The animal eats.
The dog barks.

C:\Users\Ustich\Desktop\студент\3 КУРС\Об'єктно орієнтоване програмування\на
24.10.25\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe (процесс 226
24) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рис. 2. Результат виконання програми в консолі.

У даному прикладі клас `Dog` успадковує функцію `eat()` від класу `Animal` і має власну функцію `bark()`.

2. Які ви знаєте специфікатори доступу?

У мові C++ використовуються три специфікатори доступу:

- **private** — члени класу доступні лише всередині цього класу;
- **protected** — члени класу доступні у самому класі та його похідних класах;
- **public** — члени класу доступні будь-де у програмі.

3. Що відбувається з відкритими та закритими членами базового класу, якщо базовий клас успадковується як відкритий похідним?

Якщо базовий клас успадковується як **public**, то:

- відкриті (**public**) члени залишаються відкритими;
- захищені (**protected**) залишаються захищеними;
- закриті (**private**) залишаються недоступними для похідного класу.

4. Що відбувається з закритими та відкритими членами базового класу, якщо базовий клас успадковується як закритий похідним?

Якщо базовий клас успадковується як **private**, то:

- усі відкриті (**public**) та захищені (**protected**) члени стають закритими (**private**) у похідному класі;
- закриті (**private**) члени базового класу залишаються недоступними.

5. Поясніть, навіщо потрібна категорія захищеності *protected*? Категорія *protected* використовується для створення членів класу, які:

- недоступні зовні (як `private`);
- але доступні всередині похідних класів.

Це дозволяє обмежити доступ до даних іззовні, але при цьому надає можливість похідним класам використовувати ці дані.

6. Якщо один клас успадковується іншим, яким порядок виклику конструкторів та деструкторів? Наведіть приклади.

Порядок виконання:

- **Конструктори** викликаються у порядку успадкування — спочатку базовий, потім похідний.
- **Деструктори** — у зворотному порядку: спочатку похідний, потім базовий.

Приклад:

```
#include <iostream>
using namespace std;

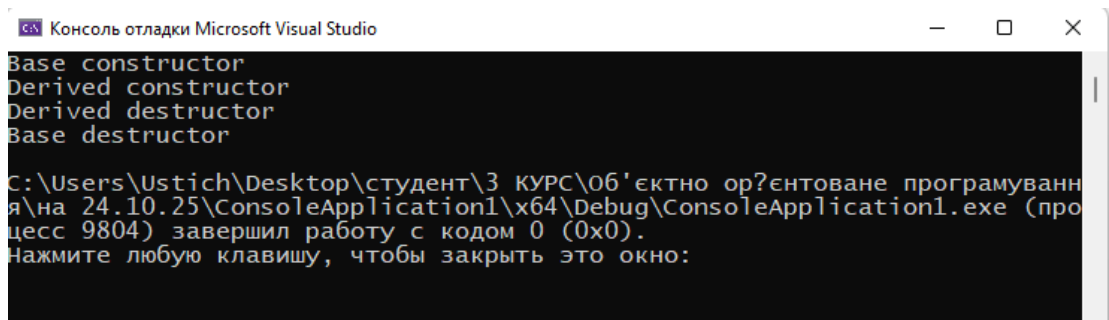
class Base {
public:
    Base() { cout << "Base constructor\n"; }
    ~Base() { cout << "Base destructor\n"; }
};

class Derived : public Base {
public:
    Derived() { cout << "Derived constructor\n"; }
    ~Derived() { cout << "Derived destructor\n"; }
};

int main() {
    Derived d;
    return 0;
}
```

Рис. 3. Код демонстрації порядку виклику конструкторів і деструкторів.

Виконання програми:



```
Консоль отладки Microsoft Visual Studio

Base constructor
Derived constructor
Derived destructor
Base destructor

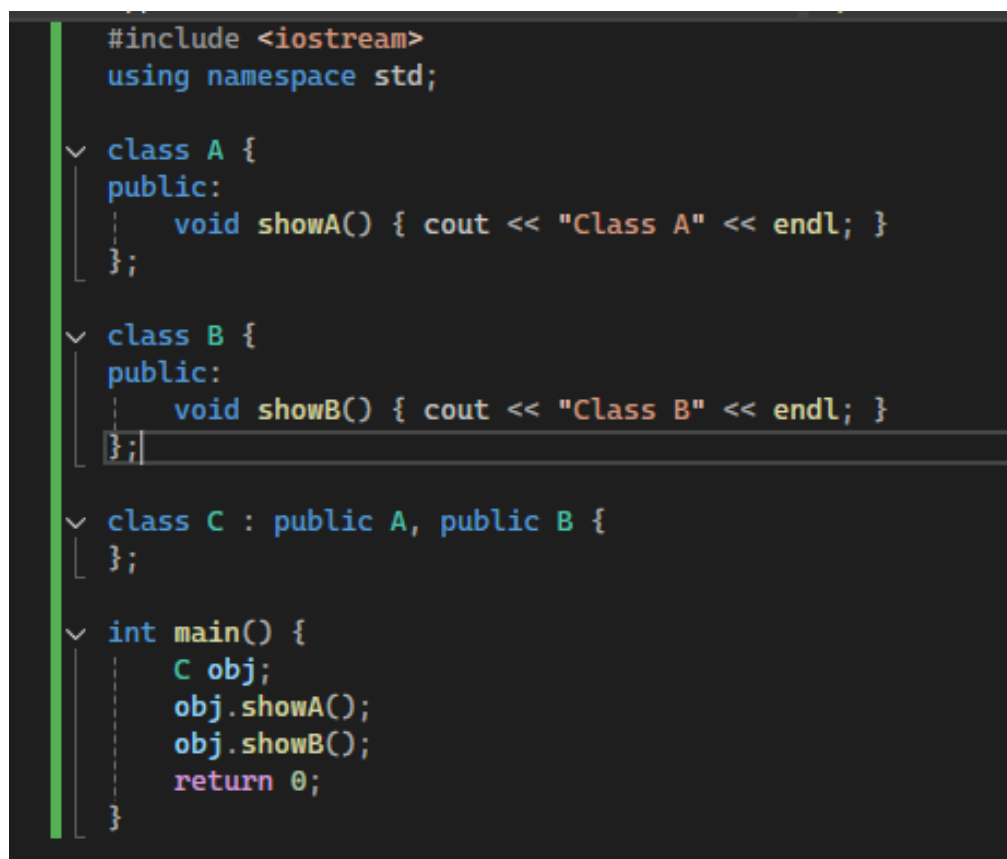
C:\Users\Ustich\Desktop\студент\3 КУРС\Об'єктно орієнтоване програмування\на 24.10.25\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe (процесс 9804) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рис. 4. Результат виконання програми в консолі.

7. Що таке множинне успадкування?

Множинне успадкування — це механізм, коли один клас успадковує властивості від кількох базових класів одночасно.

Приклад:



```
#include <iostream>
using namespace std;

class A {
public:
    void showA() { cout << "Class A" << endl; }
};

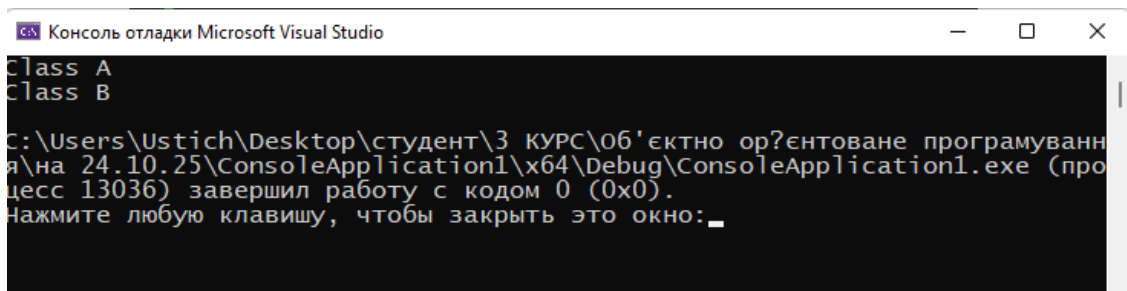
class B {
public:
    void showB() { cout << "Class B" << endl; }
};

class C : public A, public B {
};

int main() {
    C obj;
    obj.showA();
    obj.showB();
    return 0;
}
```

Рис. 5. Код прикладу множинного успадкування у Visual Studio.

Виконання програми:



```
Class A
Class B

C:\Users\Ustich\Desktop\студент\3 КУРС\Об'єктно орієнтоване програмування\на 24.10.25\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe (процесс 13036) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: _
```

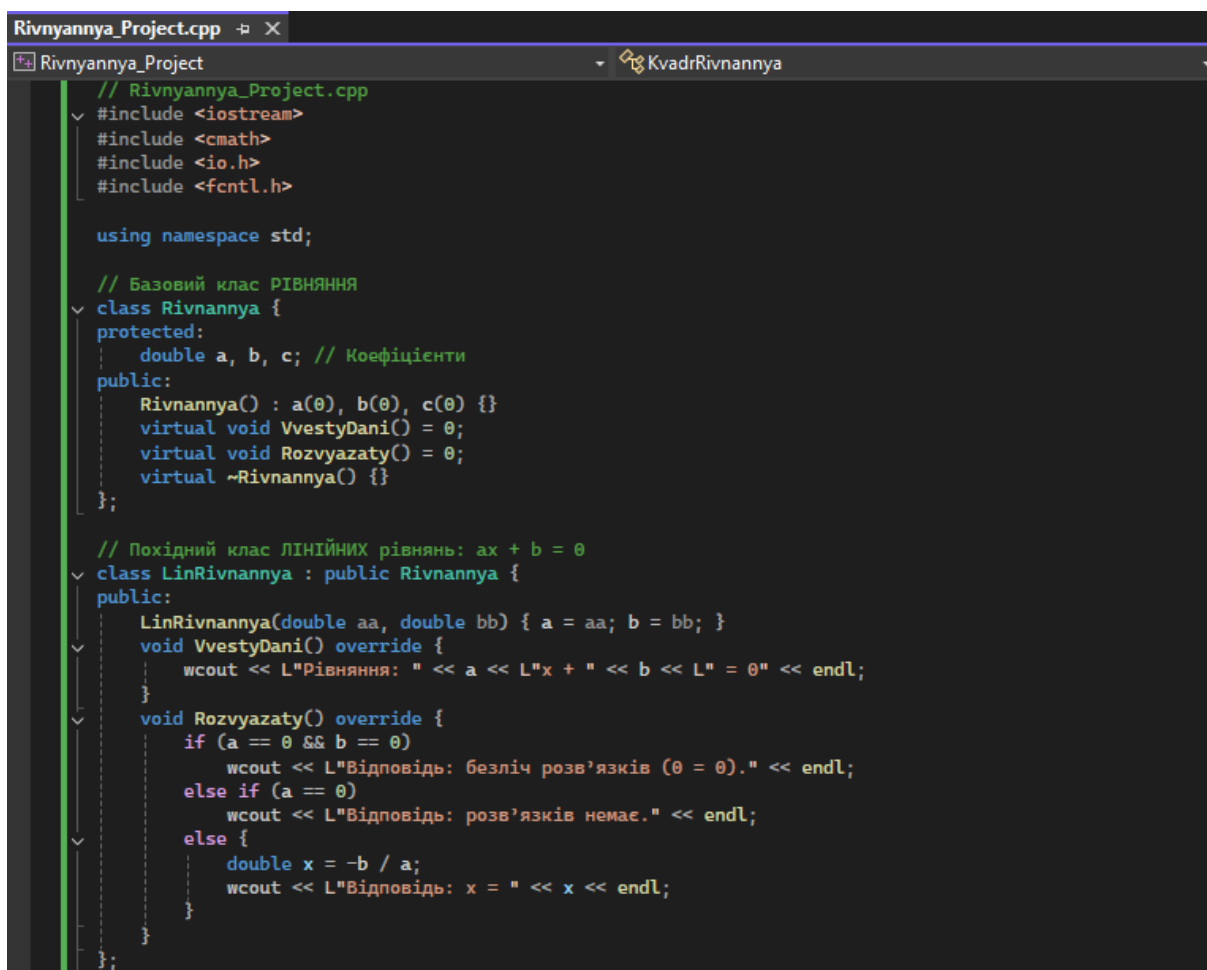
Рис. 6. Результат виконання програми.

Завдання 1

Варіант 8. Створити базовий клас РІВНЯННЯ обчислення значення x . Створити

похідні класи: клас ЛІНІЙНИХ РІВНЯНЬ ($ax+b=0$, додаються два дійсних значення - коефіцієнти a і b) і клас КВАДРАТНИХ РІВНЯНЬ ($cx^2 + a x + b = 0$, додаються три дійсних значення a, b, c).

Код програми:



```
Rivnyannya_Project.cpp
Rivnyannya_Project
KvadrRivnyannya

// Rivnyannya_Project.cpp
#include <iostream>
#include <cmath>
#include <io.h>
#include <fcntl.h>

using namespace std;

// Базовий клас РІВНЯННЯ
class Rivnyannya {
protected:
    double a, b, c; // Коефіцієнти
public:
    Rivnyannya() : a(0), b(0), c(0) {}
    virtual void VvestyDani() = 0;
    virtual void Rozvyazaty() = 0;
    virtual ~Rivnyannya() {}
};

// Похідний клас ЛІНІЙНИХ рівнянь: ax + b = 0
class LinRivnyannya : public Rivnyannya {
public:
    LinRivnyannya(double aa, double bb) { a = aa; b = bb; }
    void VvestyDani() override {
        wcout << L"Рівняння: " << a << L"x + " << b << L" = 0" << endl;
    }
    void Rozvyazaty() override {
        if (a == 0 && b == 0)
            wcout << L"Відповідь: безліч розв'язків (0 = 0)." << endl;
        else if (a == 0)
            wcout << L"Відповідь: розв'язків немає." << endl;
        else {
            double x = -b / a;
            wcout << L"Відповідь: x = " << x << endl;
        }
    }
};
```

```

// Похідний клас КВАДРАТНИХ рівнянь:  $cx^2 + ax + b = 0$ 
class KvadrRivnannya : public Rivnannya {
public:
    KvadrRivnannya(double cc, double aa, double bb) { c = cc; a = aa; b = bb; }
    void VvestyDani() override {
        wcout << L"Рівняння: " << c << L"x^2 + " << a << L"x + " << b << L" = 0" << endl;
    }
    void Rozvyazaty() override {
        if (c == 0) { // Перетворення на лінійне
            LinRivnannya temp(a, b);
            temp.VvestyDani();
            temp.Rozvyazaty();
            return;
        }
        double D = a * a - 4 * c * b;
        wcout << L"Дискримінант D = " << D << endl;
        if (D > 0) {
            double x1 = (-a + sqrt(D)) / (2 * c);
            double x2 = (-a - sqrt(D)) / (2 * c);
            wcout << L"Два розв'язки:  $x_1 =$ " << x1 << L",  $x_2 =$ " << x2 << endl;
        }
        else if (D == 0) {
            double x = -a / (2 * c);
            wcout << L"Один розв'язок:  $x =$ " << x << endl;
        }
        else
            wcout << L"Немає дійсних розв'язків ( $D < 0$ ). " << endl;
    }
};

int main() {
    // Налаштування консолі на UTF-8
    _setmode(_fileno(stdout), _O_U8TEXT);
    _setmode(_fileno(stdin), _O_U8TEXT);

    wcout << L"=== Програма для розв'язання рівнянь ===\n\n";

```

```

// Лінійні рівняння
LinRivnannya lin1(2, -4);
wcout << L"=== Розв'язання лінійного рівняння ===\n";
lin1.VvestyDani();
lin1.Rozvyazaty();
wcout << endl;

LinRivnannya lin2(0, 0);
wcout << L"=== Розв'язання лінійного рівняння ===\n";
lin2.VvestyDani();
lin2.Rozvyazaty();
wcout << endl;

// Квадратні рівняння
KvadrRivnannya kv1(1, -3, 2); //  $D > 0$ 
wcout << L"=== Розв'язання квадратного рівняння ===\n";
kv1.VvestyDani();
kv1.Rozvyazaty();
wcout << endl;

KvadrRivnannya kv2(0, 0, 0); // Некоректне
wcout << L"=== Розв'язання квадратного рівняння ===\n";
kv2.VvestyDani();
kv2.Rozvyazaty();
wcout << endl;

```

```

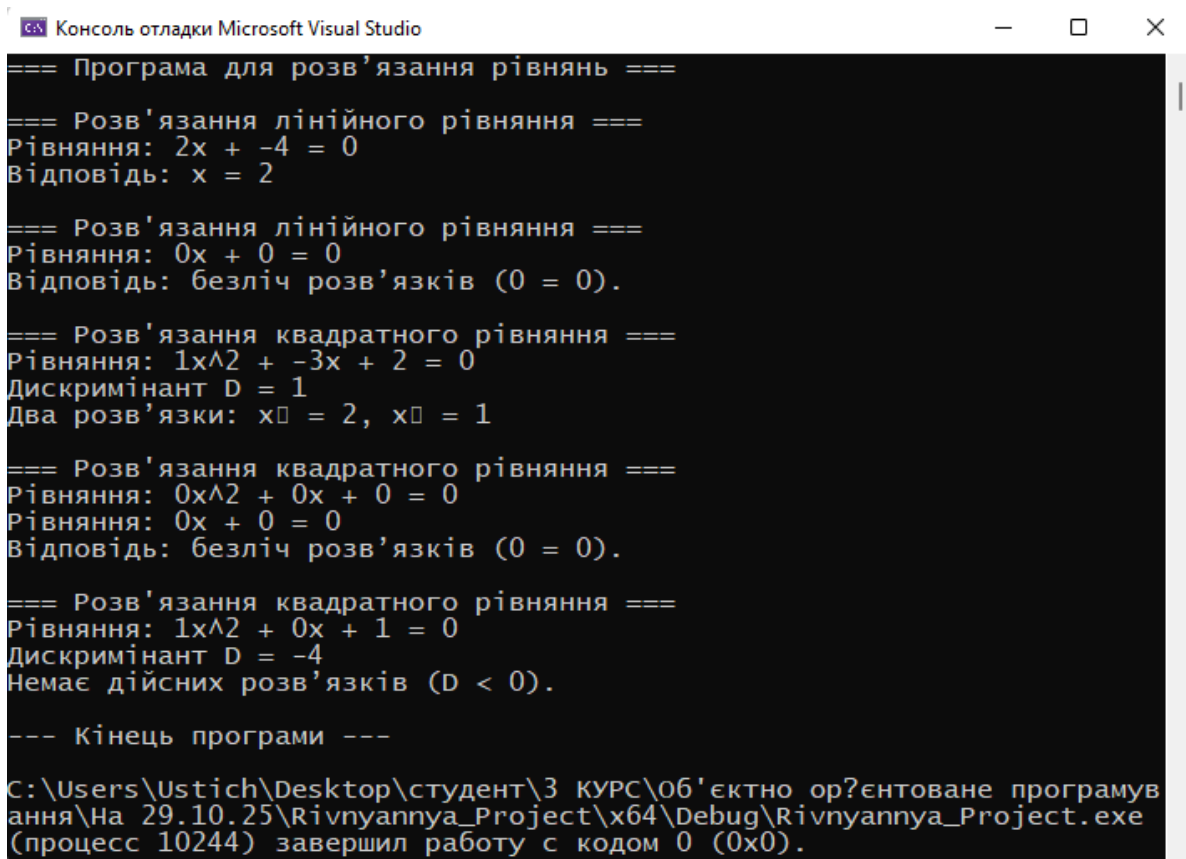
KvadrRivnannya kv3(1, 0, 1); // D < 0
wcout << L"=== Розв'язання квадратного рівняння ===\n";
kv3.VvestyDani();
kv3.Rozvyazaty();
wcout << endl;

wcout << L"--- Кінець програми ---\n";
return 0;
}

```

Рис. 7. Код програми для розв'язання лінійних та квадратних рівнянь.

Виконання програми:



```

Консоль отладки Microsoft Visual Studio
=== Програма для розв'язання рівнянь ===

=== Розв'язання лінійного рівняння ===
Рівняння: 2x + -4 = 0
Відповідь: x = 2

=== Розв'язання лінійного рівняння ===
Рівняння: 0x + 0 = 0
Відповідь: безліч розв'язків (0 = 0).

=== Розв'язання квадратного рівняння ===
Рівняння: 1x^2 + -3x + 2 = 0
Дискримінант D = 1
Два розв'язки: x1 = 2, x2 = 1

=== Розв'язання квадратного рівняння ===
Рівняння: 0x^2 + 0x + 0 = 0
Рівняння: 0x + 0 = 0
Відповідь: безліч розв'язків (0 = 0).

=== Розв'язання квадратного рівняння ===
Рівняння: 1x^2 + 0x + 1 = 0
Дискримінант D = -4
Немає дійсних розв'язків (D < 0).

--- Кінець програми ---

C:\Users\Ustich\Desktop\студент\3 КУРС\Об'єктно орієнтоване програмування\На 29.10.25\Rivnyannya_Project\x64\Debug\Rivnyannya_Project.exe
(процесс 10244) завершил работу с кодом 0 (0x0).

```

Рис. 7. Вивід програми в консолі після запуску.