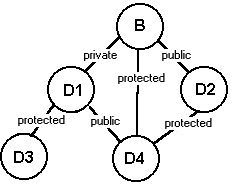
**Лабораторна робота №4**

# *Побудова множинного наслідування класів (С++) Інтерфейси – (С#)*

**Мета.** Навчитися організовувати класи та похідні класи з своїми даними та властивостями: визначати структуру класу, типи даних та методів, організовувати необхідні конструктори та деструктори, використовувати специфікатори доступу; будувати ієрархію класів, використовуючи множинне наслідування; визначати та оперувати об’єктами цих класів; отримати практичні вміння та навички проектування та побудови ієрархії класів. Отримати практичні навички використання інтерфейсів.

Варіант 7:



**Хід роботи:**

Реалізація на C++ :

#include <iostream>

#include <string.h>

#include <Windows.h>

using namespace std;

class B1

{

public:

B1()

{

cout << "Визвано конструктор В1, він без предків\n";

}

~B1()

{

cout << "Визвано деконструктор В1\n";

}

};

class D1 : private B1

{

public:

D1()

{

cout << "Визвано конструктор D1, він наслідує B1\n";

}

~D1()

{

cout << "Визвано деконструктор D1\n";

}

};

class D2 : public B1

{

public:

D2()

{

cout << "Визвано конструктор D2, він наслідує B1\n";

}

~D2()

{

cout << "Визвано деконструктор D2\n";

}

};

class D3 : protected D1

{

public: D3()

{

cout << "Визвано конструктор D3, він наслідує D1 i не наслідує B1\n";

}

~D3()

{

cout << "\nВизвано деконструктор D3\n";

}

};

class D4 : public D1, protected D2

{

public:

D4()

{

cout << "Визвано конструктор D4, він наслідує D1, D2, B1\n\n";

}

~D4()

{

cout << "\nВизвано деконструктор D4\n";

}

};

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

D4 dd4;

D3 dd3;

return 0;

}

Реалізація на C# :

using System;

namespace ООП\_4.\_1

{

class Program

{

public interface Ib

{

public void Ib\_();

}

public interface Id1

{

public void Id1\_();

}

private interface Id2

{

public void Id2\_();

}

private interface Id3

{

public void Id3\_();

}

private interface Id4

{

public void Id4\_();

}

public class D1 : Ib

{

public void Ib\_()

{

Console.WriteLine("Клас D1 наслідує інтерфейс B ");

}

}

public class D2 : Ib

{

public void Ib\_()

{

Console.WriteLine("Клас D2 наслідує інтерфейс B");

}

}

public class D3 : D1

{

public void Ib\_()

{

Console.WriteLine("Клас D3 наслідує інтерфейс B та клас D1");

}

}

public class D4 : Ib, Id1,Id2

{

public void Ib\_()

{

Console.WriteLine("Клас D4 наслідує інтерфейс B");

}

public void Id1\_()

{

Console.WriteLine("Клас D4 наслідує інтерфейс D1");

}

public void Id2\_()

{

Console.WriteLine("Клас D4 наслідує інтерфейс D2");

}

}

static void Main(string[] args)

{

D3 d3 = new D3();

d3.Ib\_();

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

D4 d4 = new D4();

d4.Ib\_();

d4.Id1\_();

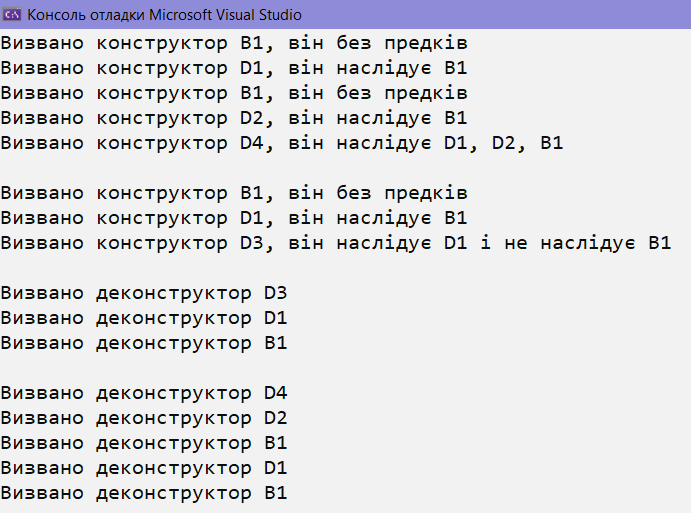
d4.Id2\_();

}

}

}

Зразок виконання:



Висновок: На цій лабораторній роботі я навчився будувати множинне успадкування класі в C++ та інтерфейсів в C#.