

Лабораторна робота № 4

УСПАДКУВАННЯ В C#.

Мета: Набути умінь і навичок створення ієрархії класів C# у середовищі Microsoft Visual Studio 2022.

Варіант 15

Завдання до лабораторної роботи

1. У консольному додатку побудувати ієрархію класів згідно варіанту.
2. Навести приклади перевантаження батьківських методів в класах-нащадках. Для цього у базового класу повинно бути 2-3 методи для перевантаження.
3. Базовий клас зробити абстрактним.
4. Продемонструвати роботу методів `virtual` та `override`.
5. Перевірити роботу модифікаторів `new` та `sealed`.
6. Створити клас, якій містить список об'єктів утворених класів. Реалізувати в ньому метод для виклику `virtual` методів об'єктів, що належать одній ієрархічній структурі. Реалізувати методи для додавання та видалення об'єктів списку.
7. Створіть абстрактний базовий клас «Фігура» з абстрактними методами для підрахунку площі і малювання фігури. Створіть похідні класи: прямокутник, коло, прямокутний трикутник зі своїми реалізаціями методу для підрахунку площі та виводу фігури на екран. Для перевірки, визначте масив посилань на абстрактний клас, за яким надаються адреси різних об'єктів класів-нащадків та продемонструйте використання віртуальних методів.

Варіант індивідуального завдання – 15. Корабель, пароплав, вітрильник, фрегат, атомохід.

Завдання 1-6:

```
using System;
```

```
using System.Collections.Generic;
```

```
abstract class Ship
```

```
{
```

```
    public abstract void Sail();
```

```
    public abstract void Stop();
```

```
}
```

```
class Steamship : Ship
```

```
{
```

```
    public override void Sail()
```

```
    {
```

```
        Console.WriteLine("Steamship sailed");
```

```
    }
```

```
    public override void Stop()
```

```
    {
```

```
        Console.WriteLine("Steamship stopped");
```

```
    }
```

```
}
```

```
class Sailboat : Ship
```

```
{
```

```
    public override void Sail()
```

```
    {
```

```
        Console.WriteLine("Sailboat sailed");
```

```
    }
```

```
public override void Stop()
{
    Console.WriteLine("Sailboat stopped");
}
}
```

```
class Frigate : Ship
{
    public override void Sail()
    {
        Console.WriteLine("Frigate sailed");
    }
}
```

```
public override void Stop()
{
    Console.WriteLine("Frigate stopped");
}
}
```

```
class Warship : Ship
{
    public override void Sail()
    {
        Console.WriteLine("Warship sailed");
    }
}
```

```
public override void Stop()
```

```
{  
    Console.WriteLine("Warship stopped");  
}  
}
```

```
class ShipCollection
```

```
{  
    private List<Ship> ships = new List<Ship>();  
  
    public void AddShip(Ship ship)  
    {  
        ships.Add(ship);  
    }  
  
    public void RemoveShip(Ship ship)  
    {  
        ships.Remove(ship);  
    }  
  
    public void CallVirtualMethod()  
    {  
        foreach (var ship in ships)  
        {  
            ship.Sail();  
            ship.Stop();  
        }  
    }  
}
```

```

class Program
{
    static void Main(string[] args)
    {
        ShipCollection collection = new ShipCollection();

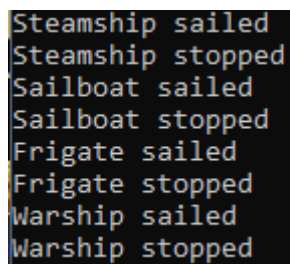
        Steamship steamship = new Steamship();
        Sailboat sailboat = new Sailboat();
        Frigate frigate = new Frigate();
        Warship warship = new Warship();

        collection.AddShip(steamship);
        collection.AddShip(sailboat);
        collection.AddShip(frigate);
        collection.AddShip(warship);

        collection.CallVirtualMethod();

        Console.ReadLine();
    }
}

```



```

Steamship sailed
Steamship stopped
Sailboat sailed
Sailboat stopped
Frigate sailed
Frigate stopped
Warship sailed
Warship stopped

```

Рис. 1 – Результат виконання пунктів 1-6

Завдання 7:

```
using System;
```

```
abstract class Figure
```

```
{  
    public abstract double CalculateArea();  
    public abstract void Draw();  
}
```

```
class Rectangle : Figure
```

```
{  
    private double length;  
    private double width;
```

```
    public Rectangle(double length, double width)
```

```
{  
    this.length = length;  
    this.width = width;  
}
```

```
    public override double CalculateArea()
```

```
{  
    return length * width;  
}
```

```
    public override void Draw()
```

```
{
```

```
        Console.WriteLine($"Rectangle with length {length} and width {width}");  
    }  
}
```

```
class Circle : Figure  
{  
    private double radius;  
  
    public Circle(double radius)  
    {  
        this.radius = radius;  
    }  
  
    public override double CalculateArea()  
    {  
        return Math.PI * radius * radius;  
    }  
  
    public override void Draw()  
    {  
        Console.WriteLine($"Circle with radius {radius}");  
    }  
}
```

```
class RightTriangle : Figure  
{  
    private double baseLength;  
    private double height;
```

```

public RightTriangle(double baseLength, double height)
{
    this.baseLength = baseLength;
    this.height = height;
}

public override double CalculateArea()
{
    return 0.5 * baseLength * height;
}

public override void Draw()
{
    Console.WriteLine($"Right triangle with base {baseLength} and height
{height}");
}
}

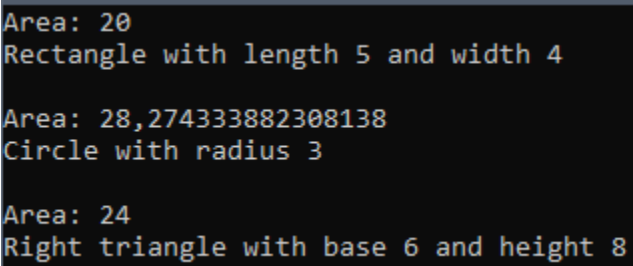
class Program
{
    static void Main(string[] args)
    {
        Figure[] figures = new Figure[3];
        figures[0] = new Rectangle(5, 4);
        figures[1] = new Circle(3);
        figures[2] = new RightTriangle(6, 8);
    }
}

```



```
foreach (var figure in figures)
{
    Console.WriteLine($"Area: {figure.CalculateArea()}");
    figure.Draw();
    Console.WriteLine();
}

Console.ReadLine();
}
}
```



```
Area: 20
Rectangle with length 5 and width 4

Area: 28,274333882308138
Circle with radius 3

Area: 24
Right triangle with base 6 and height 8
```

Рис. 2 – Результат роботи завдання із пункту 7