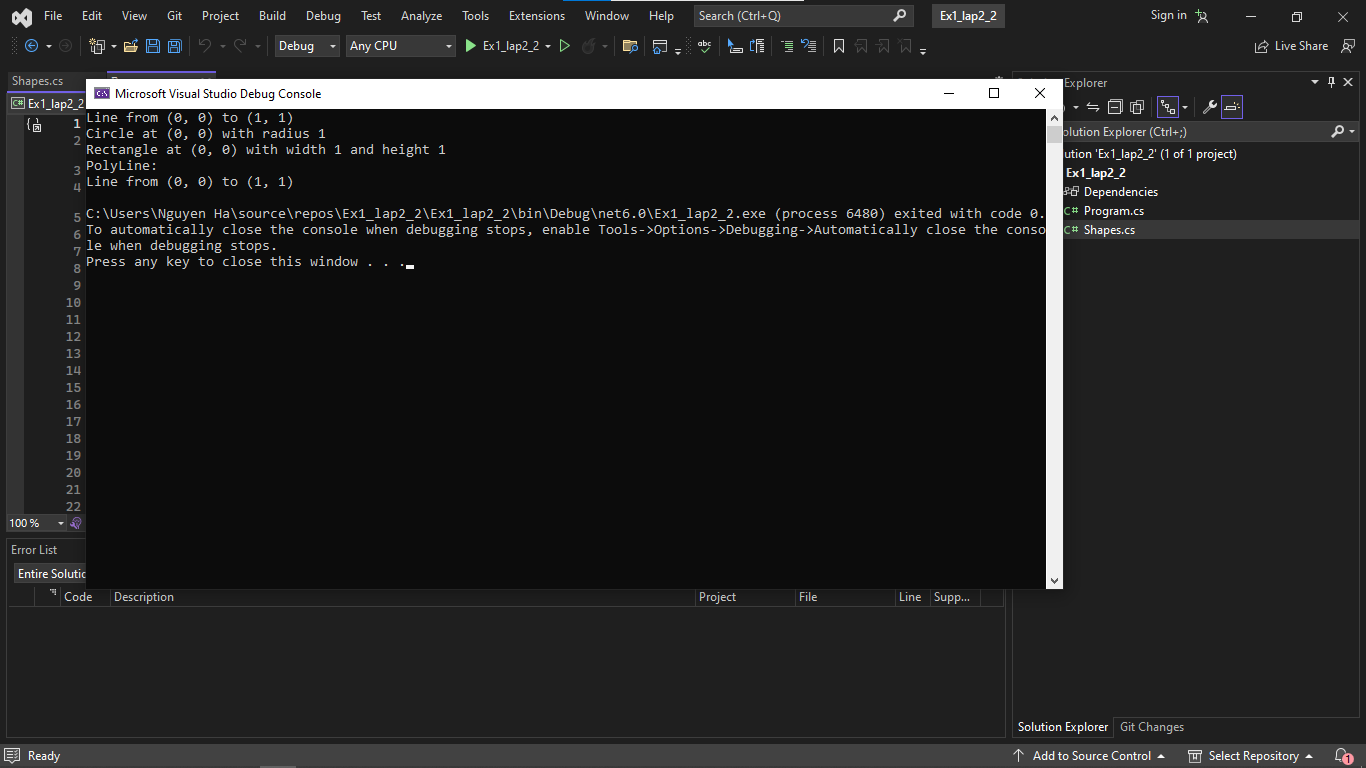
Họ và tên: Hồ Nguyên Hà . Mã SV: 20it043

Bài Tập Tuần 2(2)

**Ex1:**



Shapes.cs

using System;

namespace Shapes

{

public abstract class Shape

{

protected int x;

protected int y;

public Shape(int x, int y)

{

this.x = x;

this.y = y;

}

public void Move(int dx, int dy)

{

x += dx;

y += dy;

}

public abstract void Show();

}

public class Line : Shape

{

private int x2;

private int y2;

public Line(int x1, int y1, int x2, int y2) : base(x1, y1)

{

this.x2 = x2;

this.y2 = y2;

}

public override void Show()

{

Console.WriteLine($"Line from ({x}, {y}) to ({x2}, {y2})");

}

}

public class Circle : Shape

{

private int radius;

public Circle(int x, int y, int radius) : base(x, y)

{

this.radius = radius;

}

public override void Show()

{

Console.WriteLine($"Circle at ({x}, {y}) with radius {radius}");

}

}

public class Rectangle : Shape

{

private int width;

private int height;

public Rectangle(int x, int y, int width, int height) : base(x, y)

{

this.width = width;

this.height = height;

}

public override void Show()

{

Console.WriteLine($"Rectangle at ({x}, {y}) with width {width} and height {height}");

}

}

public class PolyLine : Shape

{

private Line[] lines;

public PolyLine(Line[] lines) : base(0, 0)

{

this.lines = lines;

}

public override void Show()

{

Console.WriteLine("PolyLine:");

foreach (var line in lines)

line.Show();

}

}

}

Program.cs

using Shapes;

class Program

{

static void Main(string[] args)

{

var line = new Line(0, 0, 1, 1);

var circle = new Circle(0, 0, 1);

var rectangle = new Rectangle(0, 0, 1, 1);

line.Show();

// Output: Line from (0, 0) to (1, 1)

circle.Show();

// Output: Circle at (0, 0) with radius 1

rectangle.Show();

// Output: Rectangle at (0, 0) with width 1 and height 1

var polyLine = new PolyLine(new Line[] { line });

polyLine.Show();

// Output:

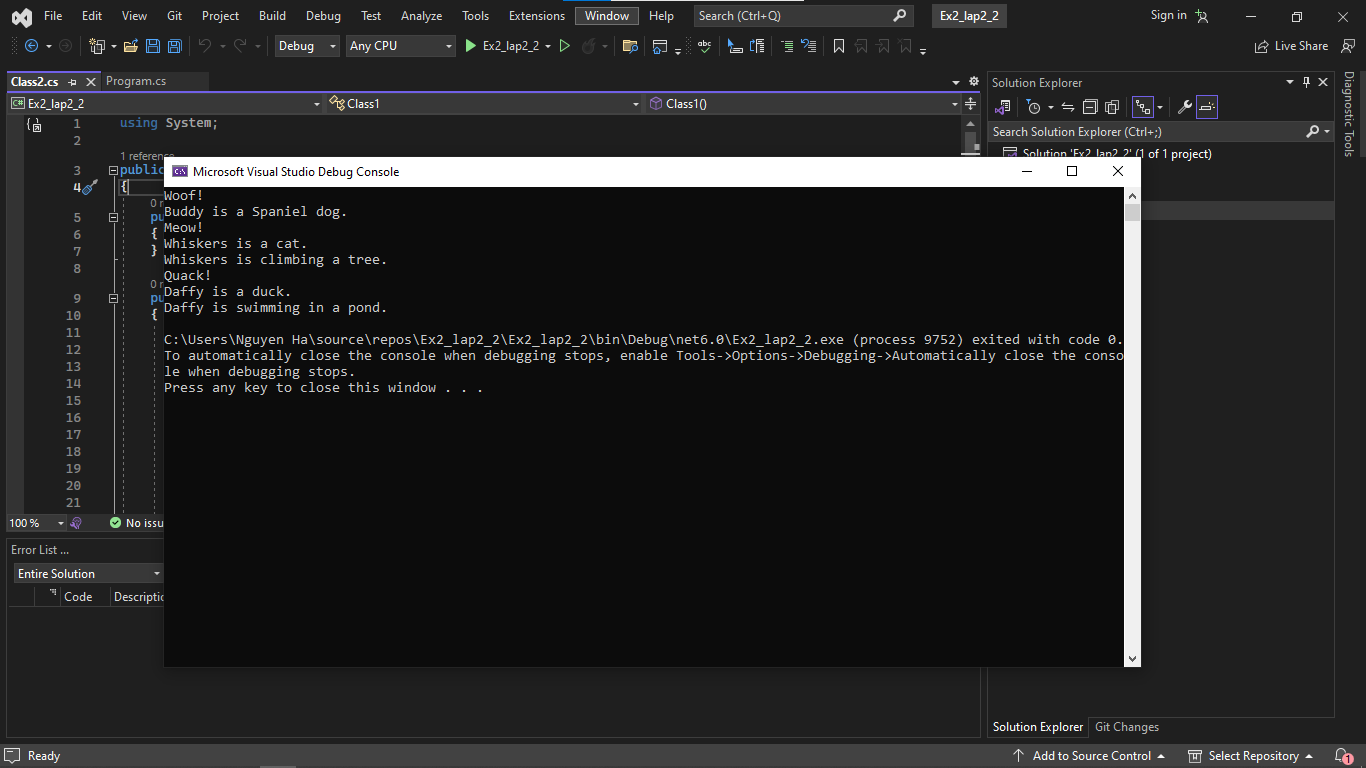
// PolyLine:

// Line from (0, 0) to (1, 1)

}

}

**Ex2:**



Program.cs

public abstract class Animal

{

public string Type { get; set; }

public Animal(string type)

{

Type = type;

}

public abstract string Sound();

public abstract string Info();

}

public class Dog : Animal

{

public string Name { get; set; }

public string Breed { get; set; }

public Dog(string name, string breed) : base("mammal")

{

Name = name;

Breed = breed;

}

public override string Sound()

{

return "Woof!";

}

public override string Info()

{

return $"{Name} is a {Breed} dog.";

}

}

public class Cat : Animal

{

public string Name { get; set; }

public Cat(string name) : base("mammal")

{

Name = name;

}

public override string Sound()

{

return "Meow!";

}

public override string Info()

{

return $"{Name} is a cat.";

}

public string Climb(string thing)

{

return $"{Name} is climbing {thing}.";

}

}

public class Duck : Animal

{

public string Name { get; set; }

public Duck(string name) : base("bird")

{

Name = name;

}

public override string Sound()

{

return "Quack!";

}

public override string Info()

{

return $"{Name} is a duck.";

}

public string Swim(string thing)

{

return $"{Name} is swimming in {thing}.";

}

}

Class2.cs

using System;

public class Class2

{

public Class2()

{

}

public static void Main(string[] args)

{

Dog dog = new Dog("Buddy", "Spaniel");

Console.WriteLine(dog.Sound());

Console.WriteLine(dog.Info());

Cat cat = new Cat("Whiskers");

Console.WriteLine(cat.Sound());

Console.WriteLine(cat.Info());

Console.WriteLine(cat.Climb("a tree"));

Duck duck = new Duck("Daffy");

Console.WriteLine(duck.Sound());

Console.WriteLine(duck.Info());

Console.WriteLine(duck.Swim("a pond"));

}

}