

Московский авиационный институт  
(национальный исследовательский университет)

## **Институт информационных технологий и прикладной математики**

Кафедра вычислительной математики и программирования

**Лабораторная работа №3-4 по курсу ООП:**

**Принцип подстановки.**

**Наследование: расширение, спецификация,  
специализация, комбинирование,  
конструирование.**

Работу выполнила:

М8О-209Б-19 Офицерова Т.И.

Группа

ФИО

Подпись

Вариант

Руководитель: \_\_\_\_\_/Кузнецова С.В./

Подпись:

Дата: 26 сентября 2020

## Специализация

### Текст программы

```
using System;

namespace lab3and4
{
    public abstract class node //абстрактный класс вершины графа
    {
        protected int size;
        protected string color;
        protected node(int size, string color)
        {
            this.size = size;
            this.color = color;
        }
        public abstract void MethodNode();//метод вершины графа
        public abstract int FunctionNode();//функция вершины графа
    }
    public class A : node
    {
        public A (int size, string color) : base(size, color) { }
        public override int FunctionNode()
        {
            return this.size;
        }
        public override void MethodNode()
        {
            Console.WriteLine(this.color);
        }
        public virtual void Method()
        {
            Console.WriteLine(" method of A ");
        }
    }
    public class B : A //наследует A
    {
        public B(int size, string color) : base(size, color) { } //конструктор
        суперкласса
        public override void Method() //переопределенный метод
        {
            Console.WriteLine(" method of B ");
        }
    }
    public class C : A //наследует A
    {
        public C(int size, string color) : base(size, color) { } //конструктор
        суперкласса
        public override void Method() //переопределенный метод
        {
            Console.WriteLine(" method of C ");
        }
    }
    public class J : A //наследует A
    {
        public J(int size, string color) : base(size, color) { } //конструктор
        суперкласса
        public override void Method() //переопределенный метод
        {
            Console.WriteLine(" method of J ");
        }
    }
    public class D : B //наследует B
    {

```

```

        public D(int size, string color) : base(size, color) { } //конструктор
        суперкласса
        public override void Method() //переопределенный метод
        {
            Console.WriteLine(" method of D ");
        }
    }
    public class F : C //наследует C
    {
        public F(int size, string color) : base(size, color) { } //конструктор
        суперкласса
        public override void Method() //переопределенный метод
        {
            Console.WriteLine(" method of F ");
        }
    }
}
class Program
{
    static void Main(string[] args)
    {
        node n= new A(1, "black"); //абстрактный класс на основе A
        Console.Write("Abstract class color: ");
        n.MethodNode();
        Console.WriteLine("Size of abstract class: {0}", n.FunctionNode());

        A a = new A(10, "white");
        a.Method(); //метод A
        Console.Write("Object A color: "); // цвет
        a.MethodNode();
        Console.WriteLine("Size of object A: {0}", a.FunctionNode()); //размер

        B b = new B(15, "blue");
        b.Method(); //метод B
        Console.Write("Object B color: "); // цвет
        b.MethodNode();
        Console.WriteLine("Size of object B: {0}", b.FunctionNode()); //размер

        C c = new C(7, "pink");
        c.Method(); //метод C
        Console.Write("Object C color: "); // цвет
        c.MethodNode();
        Console.WriteLine("Size of object C: {0}", c.FunctionNode()); //размер

        //принцип подстановки
        a = c;
        a.Method(); //метод A
        Console.Write("Object A color: "); // цвет
        a.MethodNode();
        Console.WriteLine("Size of object A: {0}", a.FunctionNode()); //размер

        J j = new J(12, "green");
        j.Method(); //метод J
        Console.Write("Object J color: "); // цвет
        j.MethodNode();
        Console.WriteLine("Size of object J: {0}", j.FunctionNode()); //размер

        D d = new D(18, "red");
        d.Method(); //метод D
        Console.Write("Object D color: "); // цвет
        d.MethodNode();
        Console.WriteLine("Size of object D: {0}", d.FunctionNode()); //размер

        F f = new F(20, "yellow");
        f.Method(); //метод F
        Console.Write("Object F color: "); // цвет
    }
}

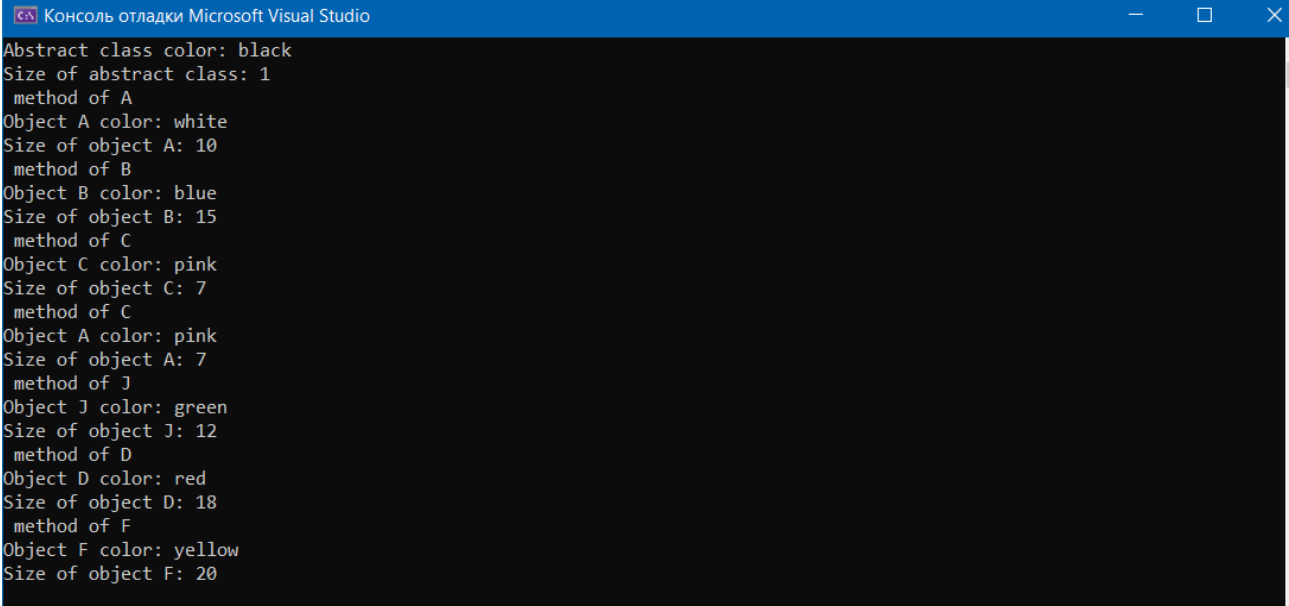
```

```

        f.MethodNode();
        Console.WriteLine("Size of object F: {0}", f.FunctionNode()); //размер
    }
}
}

```

## Результат работы



```

Консоль отладки Microsoft Visual Studio
Abstract class color: black
Size of abstract class: 1
  method of A
Object A color: white
Size of object A: 10
  method of B
Object B color: blue
Size of object B: 15
  method of C
Object C color: pink
Size of object C: 7
  method of C
Object A color: pink
Size of object A: 7
  method of J
Object J color: green
Size of object J: 12
  method of D
Object D color: red
Size of object D: 18
  method of F
Object F color: yellow
Size of object F: 20

```

## Примеры

Учитель и учитель химии, автомобиль и гоночный автомобиль, металл и щелочной металл, щелочной металл и натрий

## Спецификация

### Текст программы

```

using System;
using System.Collections.Generic;
using System.Text;

namespace lab3and4
{
    public interface Figure //интерфейс
    {
        public string TypeOfFigure(); //узнать тип фигуры
        public double PerimeterOfFigure(); //найти периметр
        public double AreaOfFigure(); //найти площадь
    }
    public class Triangle : Figure //треугольник
    {
        private double[] x;
        private double[] y;
        public Triangle(double[] x, double[] y) //треугольник конструктор
        {
            this.x = new double[3];
            this.y = new double[3];
            for (int i = 0; i < 3; i++)
            {
                this.x[i] = x[i];
                this.y[i] = y[i];
            }
        }
    }
}

```

```

public string TypeOfFigure()//тип фигуры
{
    return "triangle";
}
public double PerimeterOfFigure()//периметр
{
    double res=0;
    for (int i = 0; i < 3; i++)
    {
        for (int j = i + 1; j < 3; j++)
        {
            res += Math.Sqrt((x[j] - x[i]) * (x[j] - x[i]) + (y[j] - y[i]) *
(y[j] - y[i]));
        }
    }
    return res;
}
public double AreaOfFigure()//площадь
{
    double det = (x[0] - x[2]) * (y[1] - y[2]) - (y[0] - y[2]) * (x[1] - x[2]);
    if (det < 0)
    {
        det *= -0.5;
    }
    else
    {
        det *= 0.5;
    }
    return det;
}
public override string ToString()
{
    return $"Координаты вершин: ({this.x[0]}, {this.y[0]}), ({this.x[1]},
{this.y[1]}), ({this.x[2]}, {this.y[2]}";
}
}
public class Circle : Figure//круг
{
    private double x, y, r;
    public Circle(double x, double y, double r)//круг
    {
        this.x = x;
        this.y = y;
        this.r = r;
    }
    public string TypeOfFigure()//тип фигуры
    {
        return "circle";
    }
    public double PerimeterOfFigure()
    {
        return 2 * Math.PI * r;
    }
    public double AreaOfFigure()
    {
        return Math.PI * r * r;
    }
    public override string ToString()
    {
        return $"Центр круга: ({this.x}, {this.y}). Радиус {this.r}";
    }
}
public class Parallelogram : Figure//параллелограмм
{
    private double[] x;

```

```

private double[] y;
public Parallelogram(double[] x, double[] y)//параллелограмм
{
    this.x = new double[4];
    this.y = new double[4];
    for (int i = 0; i < 4; i++)
    {
        this.x[i] = x[i];
        this.y[i] = y[i];
    }
}
public string TypeOfFigure()//тип фигуры
{
    return "parallelogram";
}
public double PerimeterOfFigure()
{
    double res = 0;
    res += Math.Sqrt((x[2] - x[1]) * (x[2] - x[1]) + (y[2] - y[1]) * (y[2] -
y[1]));
    res += Math.Sqrt((x[1] - x[0]) * (x[1] - x[0]) + (y[1] - y[0]) * (y[1] -
y[0]));
    return res*2;
}
public double AreaOfFigure()
{
    double det = (x[0] - x[2]) * (y[1] - y[2]) - (y[0] - y[2]) * (x[1] - x[2]);
    if (det < 0)
    {
        det *= -0.5;
    }
    else
    {
        det *= 0.5;
    }
    return det*2;
}
public override string ToString()
{
    return $"Координаты вершин: ({this.x[0]}, {this.y[0]}), ({this.x[1]},
{this.y[1]}), ({this.x[2]}, {this.y[2]}), ({this.x[3]}, {this.y[3]}));";
}
}
class Program
{
    static void Main(string[] args)
    {
        double[] x = new double[3] { 0, 0, 1 };
        double[] y = new double[3] { 0, 1, 0 };
        Figure t = new Triangle(x, y);//подстановка на основе интерфейса->треугольник
        Console.WriteLine("Тип фигуры: {0}", t.TypeOfFigure());
        Console.WriteLine("Периметр фигуры(треугольника): {0}",
t.PerimeterOfFigure());
        Console.WriteLine("Площадь фигуры(треугольника): {0}", t.AreaOfFigure());
        Console.WriteLine(t);//координаты вершин треугольника
        t = new Circle(0, 0, 1);//круг
        Console.WriteLine("Тип фигуры: {0}", t.TypeOfFigure());
        Console.WriteLine("Периметр фигуры(круга): {0}", t.PerimeterOfFigure());
        Console.WriteLine("Площадь фигуры(круга): {0}", t.AreaOfFigure());
        Console.WriteLine(t);
        double[] x1 = new double[4] { 2, 3, 7, 6 };
        double[] y1 = new double[4] { 0, 3, 3, 0 };
        t = new Parallelogram(x1, y1);//параллелограмм
        Console.WriteLine("Тип фигуры: {0}", t.TypeOfFigure());
    }
}

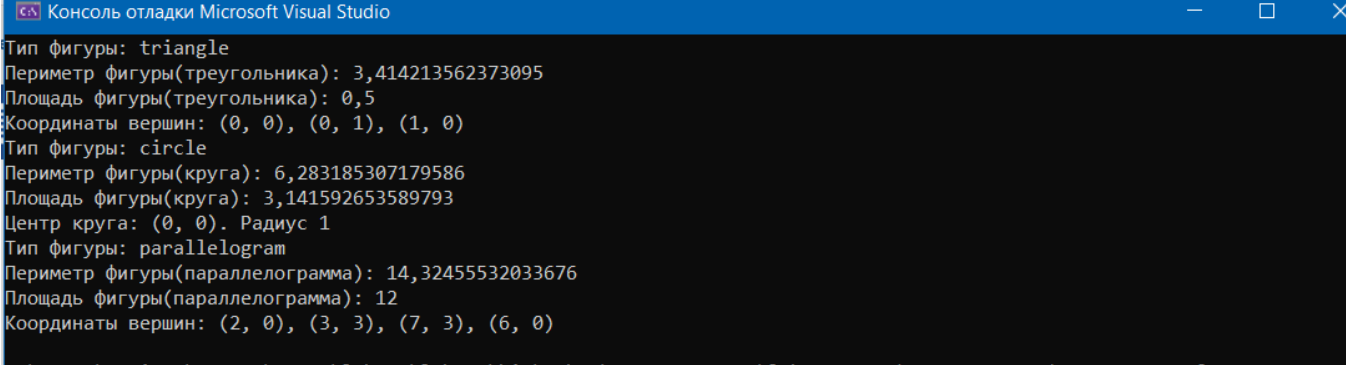
```

```

        Console.WriteLine("Периметр фигуры(параллелограмма): {0}",
t.PerimeterOfFigure());
        Console.WriteLine("Площадь фигуры(параллелограмма): {0}", t.AreaOfFigure());
        Console.WriteLine(t);
    }
}
}

```

## Результат работы



```

Консоль отладки Microsoft Visual Studio
Тип фигуры: triangle
Периметр фигуры(треугольника): 3,414213562373095
Площадь фигуры(треугольника): 0,5
Координаты вершин: (0, 0), (0, 1), (1, 0)
Тип фигуры: circle
Периметр фигуры(круга): 6,283185307179586
Площадь фигуры(круга): 3,141592653589793
Центр круга: (0, 0). Радиус 1
Тип фигуры: parallelogram
Периметр фигуры(параллелограмма): 14,32455532033676
Площадь фигуры(параллелограмма): 12
Координаты вершин: (2, 0), (3, 3), (7, 3), (6, 0)

```

## Примеры

Интерфейс SortedSet и его реализации (в C#), структура элемента периодической таблицы Менделеева и любой элемент.

## Вывод

Указателю суперкласса можно присваивать любой объект класса-наследника и работать с ним как с объектом супер-класса.

Специализация позволяет описывать более конкретные случаи некоторого класса, переопределяя часть его поведения. Спецификация создает некоторую основу с помощью абстрактного супер-класса или интерфейса, и уже на основе этой базы пишутся остальные классы.