

Министерство науки и высшего образования Российской Федерации  
Муромский институт (филиал)  
Федерального государственного бюджетного образовательного учреждения  
высшего образования  
«Владимирский государственный университет  
имени Александра Григорьевича и Николая Григорьевича Столетовых»

Факультет \_\_\_\_\_ ИТР \_\_\_\_\_

Кафедра \_\_\_\_\_ ПИИ \_\_\_\_\_

## **ЛАБОРАТОРНАЯ РАБОТА №1**

По Объектно-ориентированному программированию

Руководитель

Привезенцев Д.Г.

(фамилия, инициалы)

(подпись)

(дата)

Студент ПИИ - 121

(группа)

Ермилов М.В.

(фамилия, инициалы)

(подпись)

(дата)

Муром 2022

## Лабораторная работа №1

**Тема:** Полиморфизм методов класса

Ход работы:

**Задание 1:** Построить класс 1-го уровня с указанными в индивидуальном задании (табл. 1) полями и методами:

- конструктор;
- функция, которая определяет «качество» объекта – Q по заданной формуле (табл. 1, столб. 2);
- вывод информации об объекте.

Построить класс 2-го уровня (класс-потомок), который содержит:

- дополнительное поле P;
- функция, которая определяет «качество» объекта класса 2-го уровня – Qp, которая перекрывает функцию качества класса 1-го уровня (Q), выполняя вычисление по новой формуле (табл. 1, столб. 3).

Создать проект для демонстрации работы: ввод и вывод информации об объектах классов 1-го и 2-го уровней.

№ вар.	Поля и функция «качества» (Q ) класса 1-го уровня	Поле и функция «качества» Qp класса 2-го уровня
4	Кабель: - тип; - количество жил кабеля; - диаметр. - $Q = \text{диаметр} / \text{количество жил}$	P: наличие оплетки Qp: если P - истина, то $Qp = 2 \cdot Q$ ; иначе $Qp = 0,7 \cdot Q$

Таблица 1 - Задание.

					МИ ВлГУ 09.03.04						
Изм.	Лист	№ докум.	Подпись	Дата	Полиморфизм Методов класса			Лит.	Лист	Листов	
Разраб.		Ермилов М.В.									
Провер.		Привезенцев Д.Г.									
Реценз.											
Н. Контр.											
Утверд.											
					ПИН-121						

Код по заданию 1:

Класс 1-го уровня:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

class Wire
{
    /// <summary>
    /// Кол-во жил в кабеле
    /// </summary>
    public int CountCores;
    /// <summary>
    /// Диаметр кабеля в мм
    /// </summary>
    public double D;
    /// <summary>
    /// Тип кабеля
    /// </summary>
    public string Type;
    /// <summary>
    /// Качество объекта
    /// </summary>
    public double Q { get { return D / CountCores; } }

    /// <summary>
    /// Конструктор
    /// </summary>
    /// <param name="Type">Тип кабеля</param>
    /// <param name="CountCores">Кол-во жил в кабеле</param>
    /// <param name="D">Диаметр кабеля в мм</param>
    public Wire(string Type, int CountCores, double D)
    {
        this.CountCores = CountCores;
        this.D = D;
        this.Type = Type;
    }

    public override string ToString()
    {
        return $"Тип: {Type};\nКол-во жил: {CountCores};\nДиаметр: {D}мм;\nQ: {Q}";
    }
}
```

Класс 2-го уровня:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

class WirePlus : Wire
{
    /// <summary>
    /// Наличие оплетки на кабеле
    /// </summary>
    public bool P;
```

					МИ ВлГУ 09.03.04	Лист
						3
Изм.	Лист	№ докум.	Подпись	Дата		

```

    /// <summary>
    /// Качество объекта
    /// </summary>
    public double Qp { get { if (P) return 2 * Q; return 0.7 * Q; } }
    /// <summary>
    /// Конструктор
    /// </summary>
    /// <param name="Type">Тип кабеля</param>
    /// <param name="CountCores">Кол-во жил в кабеле</param>
    /// <param name="D">Диаметр кабеля в мм</param>
    /// <param name="P">Наличие оплетки на кабеле</param>
    public WirePlus(string Type, int CountCores, double D, bool P) : base(Type,
CountCores, D)
    {
        this.P = P;
    }
    public override string ToString()
    {
        return $"{base.ToString()}\nНаличие оплетки: {(P ? "+" : "-")};\nQp: {Qp}";
    }
}

```

Основной класс для вызова и демонстрации работы двух предыдущих:

```

using System;

namespace lab_1
{
    class Program
    {
        static Random R = new Random(100);
        static void Main(string[] args)
        {
            WiteTest(3);
            WirePlusTest(3);
        }
        static void WiteTest(int I)
        {
            Console.WriteLine("\n===== \nТест класса
Wire:\n===== \n");
            for (int i = 0; i < I; i++)
            {
                double b = (double)R.Next(1, 21) / R.Next(1, 5);
                Wire a = new Wire(WireTypeGenerate(), R.Next(1, 21), b);
                Console.WriteLine($" \nПровод #{(i+1)}:");
                Console.WriteLine(a);
            }
        }
        static void WirePlusTest(int I)
        {
            Console.WriteLine("\n===== \nТест класса
WirePlus:\n===== \n");
            for (int i = 0; i < I; i++)
            {
                bool b = R.Next(0, 2) == 1 ? true : false;
                double c = (double)R.Next(1, 21) / R.Next(1, 5);
                WirePlus a = new WirePlus(WireTypeGenerate(), R.Next(1, 21), c, b);
                Console.WriteLine($" \nПровод #{(i + 1)}:");
                Console.WriteLine(a);
            }
        }
        static string WireTypeGenerate()
        {
            return (Char.ConvertFromUtf32(R.Next(97, 103)) +
Char.ConvertFromUtf32(R.Next(97, 103))) + "-" + R.Next(100, 1000).ToString();
        }
    }
}

```

					МИ ВлГУ 09.03.04	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

```

}
}
}

```

```

=====
Тест класса Wire:
=====

Провод №1:
Тип: ef-419;
Кол-во жил: 19;
Диаметр: 20мм;
Q: 1,0526315789473684;

Провод №2:
Тип: са-968;
Кол-во жил: 11;
Диаметр: 5мм;
Q: 0,45454545454545453;

Провод №3:
Тип: fc-908;
Кол-во жил: 20;
Диаметр: 4мм;
Q: 0,2;

```

Рисунок 1 - пример работы класса 1-го уровня.

```

=====
Тест класса WirePlus:
=====

Провод №1:
Тип: cf-970;
Кол-во жил: 11;
Диаметр: 1мм;
Q: 0,09090909090909091;
Наличие оплетки: -;
Qr: 0,06363636363636363;

Провод №2:
Тип: bd-194;
Кол-во жил: 16;
Диаметр: 6,333333333333333мм;
Q: 0,3958333333333333;
Наличие оплетки: +;
Qr: 0,7916666666666666;

Провод №3:
Тип: af-577;
Кол-во жил: 10;
Диаметр: 3,3333333333333335мм;
Q: 0,33333333333333337;
Наличие оплетки: +;
Qr: 0,6666666666666667;

```

Рисунок 2 - пример работы класса 2-го уровня.

					МИ ВлГУ 09.03.04	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

**Задание 1:** написать программу (по технологии Windows Forms) согласно заданию (см. табл. 2). Во всех классах описать необходимые конструкторы, при помощи которых будут создаваться объекты классов. Параметры создаваемых объектов вводить с клавиатуры в поле формы и передавать в конструкторы объектов в виде параметров. Вывод информации должен осуществляться в многострочное текстовое поле (TextBox).

№ вар.	Задача
4, 14, 24	<p>Создать класс Прямоугольник, заданный значениями длин двух сторон (a и b), с виртуальными методами «Периметр» и «Площадь», возвращающими периметр и площадь соответственно, а также виртуальный метод «Увеличить в два раза», увеличивающий в два раза каждую из сторон. Определить также метод «Информация», который возвращает строку, содержащую информацию об треугольнике: длины сторон, периметр и площадь .</p> <p>Создать также класс наследник Прямоугольник со скругленными углами, с дополнительным параметром радиус скругления (r). Для него переопределить. Периметр по формуле <math>p = 8 \cdot r + 2 \cdot \pi \cdot r</math>, где p – периметр обычного прямоугольника с теми же сторонами, а Площадь по формуле <math>S = 4 \cdot r^2 + \pi \cdot r^2</math>, где S – площадь обычного прямоугольника. Также переопределить метод «Увеличить в два раза» так, чтобы он также увеличивал в два раза радиус скругления (по-прежнему увеличивая стороны в два раза). В главной программе по нажатию на кнопку создать обычный прямоугольник и прямоугольник со скругленными углами и вывести информацию о них. После этого увеличить оба прямоугольника в два раза и выдать обновленную информацию.</p>

Таблица 2 - Задание.

					МИ ВлГУ 09.03.04	Лист
Изм.	Лист	№ докум.	Подпись	Дата		6

Код по заданию 2:

Код класса прямоугольника:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

class Rectangle
{
    private protected int _width;
    private protected int _height;

    /// <summary>
    /// Ширина прямоугольника
    /// </summary>
    public virtual int Width
    {
        get { return _width; }
        set { _width = Math.Abs(value); }
    }

    /// <summary>
    /// Высота прямоугольника
    /// </summary>
    public virtual int Height
    {
        get { return _height; }
        set { _height = Math.Abs(value); }
    }

    /// <summary>
    /// Периметр прямоугольника
    /// </summary>
    public virtual double Perimeter { get { return Width * 2 + Height * 2; } }

    /// <summary>
    /// Площадь прямоугольника
    /// </summary>
    public virtual double Square { get { return Width * Height; } }

    /// <summary>
    /// Конструктор
    /// </summary>
    /// <param name="Width">Ширина прямоугольника</param>
    /// <param name="Height">Высота прямоугольника</param>
    public Rectangle(int Width, int Height)
    {
        this.Height = Height;
        this.Width = Width;
    }

    /// <summary>
    /// Метод увеличивающий все размеры в 2 раза
    /// </summary>
    public virtual void upSize()
    {
        Width *= 2;
        Height *= 2;
    }
}
```

					МИ ВлГУ 09.03.04	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

```

    /// <summary>
    /// Текстовая информация об объекте
    /// </summary>
    /// <returns>Возвращает все параметры с их описанием в текстовом формате</returns>
    public string Info() { return ToString(); }

    public override string ToString() { return $"Высота: {Height};\nШирина:
{Width};\nПериметр: {Perimeter};\nПлощадь: {Square};\n"; }
}

```

Код класса прямоугольника с закруглёнными краями:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

class RectangleRounded : Rectangle
{
    private protected int _radius;

    /// <summary>
    /// Радиус углов прямоугольника
    /// </summary>
    public virtual int Radius
    {
        get { return _radius; }
        set
        {
            int v = Math.Abs(value) * 2;
            if (v > _height || v > _width)
            {
                if(v > _width)
                    _radius = _width / 2;
                if (v > _height)
                    _radius = _height / 2;
            }
            else
                _radius = Math.Abs(value);
        }
    }

    /// <summary>
    ///
    /// </summary>
    /// <param name="Width">Ширина прямоугольника</param>
    /// <param name="Height">Высота прямоугольника</param>
    /// <param name="Radius">Радиус углов прямоугольника</param>
    public RectangleRounded(int Width, int Height, int Radius) : base(Width, Height)
    {
        this.Radius = Radius;
    }

    public override int Height
    {
        get => base.Height;
        set
        {

```

					МИ ВлГУ 09.03.04	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		



```

        int v = Math.Abs(value);
        if (_radius * 2 > v)
        {
            v = _radius * 2;
        }
        _height = v;
    }
}
public override int Width
{
    get => base.Width;
    set
    {
        int v = Math.Abs(value);
        if (_radius * 2 > v)
        {
            v = _radius * 2;
        }
        _width = v;
    }
}
public override double Square { get { return base.Square - 4 * Math.Pow(Radius, 2)
+ Math.PI * Math.Pow(Radius, 2); } }
public override double Perimeter { get { return base.Perimeter - 8 * Radius + 2 *
Math.PI * Radius; } }
public override void upSize()
{
    base.upSize();
    Radius *= 2;
}
public override string ToString() { return $"Радиус угла: {Radius};\n" +
base.ToString(); }
}

```

Код класса определяющий функционал приложения:

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace lab_1_2
{
    public partial class Form1 : Form
    {
        Rectangle rectangle;

        public Form1()
        {
            InitializeComponent();
        }
        private void buttonFunctionGenerate(object sender, EventArgs e)
        {
            int width = 0, height = 0;
            int.TryParse(textBoxWidth.Text, out width);
            int.TryParse(textBoxHeight.Text, out height);
            if (checkBoxAngle.Checked)

```

```

        {
            int radius = 0;
            int.TryParse(textBoxRadius.Text, out radius);
            rectangle = new RectangleRounded(width, height, radius);
        }
        else
        {
            rectangle = new Rectangle(width, height);
            UpdateText();
        }
    private void buttonFunctionUpsize(object sender, EventArgs e)
    {
        rectangle.upSize();
        UpdateText();
    }
    private void UpdateText()
    {
        LabelTextInfo.Text = rectangle.ToString();
    }
}
}

```

Код основного класса, который запускает всё:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace lab_1_2
{
    static class Program
    {
        /// <summary>
        /// Главная точка входа для приложения.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}

```

Form1

Высота

Ширина

Скругление на углах ☐

Радиус

Сгенерировать

Увеличить в 2 раза

Рисунок 3 - Запуск приложения.

Form1

Высота

Ширина

Скругление на углах ☐

Радиус

Высота: 10;  
Ширина: 10;  
Периметр: 40;  
Площадь: 100;

Сгенерировать

Увеличить в 2 раза

Рисунок 4 - Ввод данных (ширина и высота 10) и генерация прямоугольника.

Form1

Высота

Ширина

Скругление на углах ☐

Радиус

Высота: 20;  
Ширина: 20;  
Периметр: 80;  
Площадь: 400;

Сгенерировать

Увеличить в 2 раза

Рисунок 5 - Увеличение данных прямоугольника в 2 раза.

Рисунок 6 - Попытка сгенерировать прямоугольник с введенными данными радиуса без пометки, что данный прямоугольник имеет скругленные углы.

Рисунок 7 - Сгенерированный прямоугольник со скругленными углами.

Примечание по 2 заданию:

В классах `Rectangle` и `RectangleRounded`, есть публичные свойства `Width`, `Height` и `Radius`. Они нужны для того, чтобы поправить ввод сведений о прямоугольнике, к примеру, что радиус не должен быть больше половины ширины и длины, или что ширина, длина и радиус должны быть положительными величинами.