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	ПИн	

ЛАБОРАТОРНАЯ РАБОТА №1

По Объектно-ориентированному программированию

Руководитель					
Привезенцев	з Д.Г.				
(фамилия, иници	іалы)				
(подпись)	(дата)				
Студент <u>ПИн - 121</u> (группа)					
Ермилов М.В.					
(фамилия, иници	алы)				
(подпись)	(дата)				

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 Р;
- 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)	Поле и функция «качества» Qp класса
	класса 1-го уровня	2-го уровня
4	Кабель:	Р: наличие оплетки
	- тип;	Qp: если P - истина,
	- количество жил кабеля;	то $Qp = 2 \cdot Q$;
	- диаметр.	иначе $Qp = 0.7 \cdot Q$
	- Q = диаметр / количество жил	

Таблица 1 - Задание.

					МИ ВлГУ 09.03.04			
Изм.	Лист	№ докум.	Подпись	Дата				
Разр	аб.	Ермилов М.В.			Полиморфизм	Лит.	Лист	Листов
Пров	ер.	Привезенцев Д.Г.			Методов		2	12
Реце	Н3.				, ,	ПИн-121		
Н. Кс	нтр.				класса			
Утве	₽р∂.							

```
Код по заданию 1:
Класс 1-го уровня:
using System;
using System.Collections.Generic;
using System.Ling;
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.Ling;
using System.Text;
using System.Threading.Tasks;
class WirePlus : Wire
{
    /// <summary>
    /// Наличие оплетки на кабеле
    /// </summary>
    public bool P;
                                                                                         Лист
```

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 $"{base.ToString()}\nНаличие оплетки: {(P ? "+" : "-")};\nQp: {Qp};";
   }
}
Основной класс для вызова и демонстрации работы двух предыдущих:
using System;
namespace lab_1
   class Program
       static Random R = new Random(100);
       static void Main(string[] args)
       {
           WiteTest(3);
           WirePlusTest(3);
       static void WiteTest(int I)
           Console.WriteLine("\n=========\nTecт класса
Wire:\n====
                              =====\n");
           for (int i = 0; i < I; i++)</pre>
               double b = (double)R.Next(1, 21) / R.Next(1, 5);
               Wire a = new Wire(WireTypeGenerate(), R.Next(1, 21), b);
               Console.WriteLine($"\пПровод №{(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 ७{(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();
                                                                                    Лист
```

```
}
   }
}
                         ______
                        ест 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: -;
Qp: 0,06363636363636363;
Провод №2:
Тип: bd-194;
Кол-во жил: 16;
Диаметр: 6,3333333333333333
Q: 0,3958333333333333;
Наличие оплетки: +;
Qp: 0,791666666666666;
Провод №3:
Тип: af-577;
Кол-во жил: 10;
Диаметр: 3,3333333333333335мм;
Q: 0,33333333333333333;
Наличие оплетки: +;
Qp: 0,666666666666667;
```

Рисунок 2 - пример работы класса 2-го уровня.

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 d поля формы и передавать в конструкторы объектов в виде параметров. Вывод информации должен осуществляться в многострочное текстовое поле (TextBox).

№ вар.	Задача
4, 14, 24	Создать класс Прямоугольник, заданный значениями длин двух
	сторон (а 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.
	Создать также класс наследник Прямоугольник со скругленными
	углами, с дополнительным параметром радиус скругления (r). Для
	него переопределить. Периметр по формуле р $-8 \cdot r + 2 \cdot \pi \cdot r$, где р $-$
	периметр обычного прямоугольника с теми же сторонами, а
	Площадь по формуле $S-4\cdot r^2+\pi\cdot r^2$, 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.

Таблица 2 - Задание.

Изм.	Лист	№ докум.	Подпись	Дата

```
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;
    }
```

```
/// <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};\пШирина:
{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
```

```
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 (\text{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 buttomFunctionGenerate(object sender, EventArgs e)
            int width = 0, height = 0;
            int.TryParse(textBoxWidth.Text, out width);
            int.TryParse(TextBoxHeight.Text, out height);
            if (checkBoxAngle.Checked)
```

Лист

№ докум.

Подпись

```
{
                int radius = 0;
                int.TryParse(textBoxRadius.Text, out radius);
                rectangle = new RectangleRounded(width, height, radius);
            }
            else
                rectangle = new Rectangle(width, height);
            UpdateText();
        private void buttomFunctionUpsize(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());
        }
    }
}
```

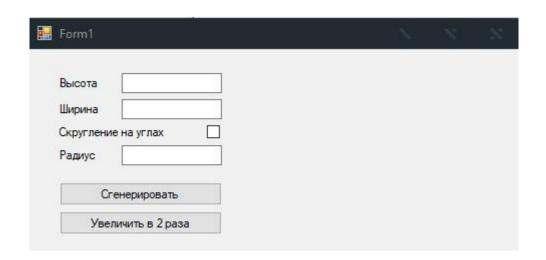


Рисунок 3 - Запуск приложения.

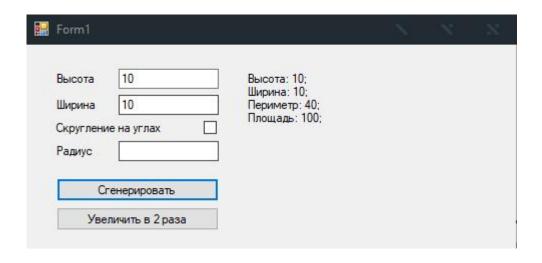


Рисунок 4 - Ввод данных (ширина и высота 10) и генерация прямоугольника.

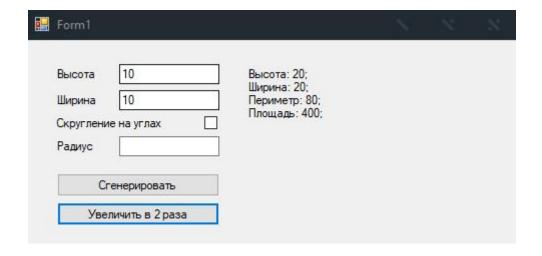


Рисунок 5 - Увеличение данных прямоугольника в 2 раза.

Изм.	Лист	№ докум.	Подпись	Дата

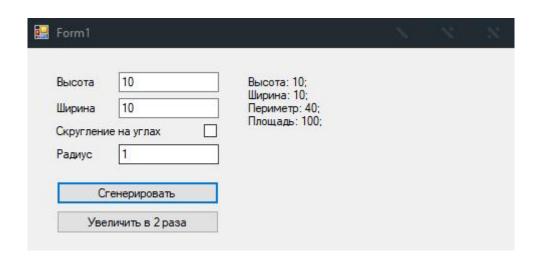


Рисунок 6 - Попытка сгенерировать прямоугольник с введенными данными радиуса без пометки, что данный прямоугольник имеет скругленные углы.

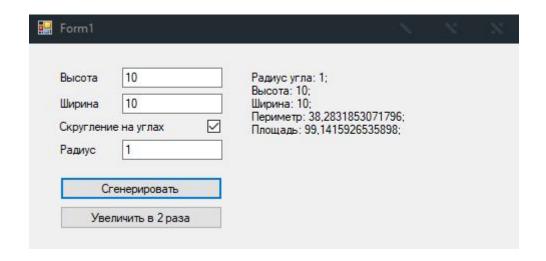


Рисунок 7 - Сгенерированный прямоугольник со скругленными углами.

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 длинны, или что ширина, длинна и радиус должны быть положительными величинами.

Изм.	Лист	№ докум.	Подпись	Дата