**Отчет по Лабораторной работе №2**

Студентка группы ИУ5-34

Лескина Виктория

Дата 04.10.17

**Формулировка задачи:**

Разработать программу, реализующую работу с классами.

1. Программа должна быть разработана в виде консольного приложения на

языке C#.

2. Абстрактный класс «Геометрическая фигура» содержит виртуальный метод

для вычисления площади фигуры.

3. Класс «Прямоугольник» наследуется от класса «Геометрическая фигура».

Ширина и высота объявляются как свойства (property). Класс должен

содержать конструктор по параметрам «ширина» и «высота».

4. Класс «Квадрат» наследуется от класса «Прямоугольник». Класс должен

содержать конструктор по длине стороны.

5. Класс «Круг» наследуется от класса «Геометрическая фигура». Радиус

объявляется как свойство (property). Класс должен содержать конструктор

по параметру «радиус».

6. Для классов «Прямоугольник», «Квадрат», «Круг» переопределить

виртуальный метод Object.ToString(), который возвращает в виде строки

основные параметры фигуры и ее площадь.

7. Разработать интерфейс IPrint. Интерфейс содержит метод Print(), который

не принимает параметров и возвращает void. Для классов «Прямоугольник»,

«Квадрат», «Круг» реализовать наследование от интерфейса IPrint.

Переопределяемый метод Print() выводит на консоль информацию,

возвращаемую переопределенным методом ToString().

**Код**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab2

{

/// <summary>

/// Abstract class

/// </summary>

abstract class GeomFigure

{

public abstract double Area();

}

interface IPrint

{

void Print();

}

class Rectangle : GeomFigure, IPrint

{

//------------------properties------------------------------

private double \_width; //data storage for properties

public double width //definition

{

get { return \_width; }

set { \_width = value; }

}

public double height { get; set; } //autodefinition

//----------------------------------------------------------

public Rectangle(double w, double h) { \_width = w; height = h; } //constructor

public override double Area()

{

return \_width \* height;

}

public override string ToString()

{

return "Rectangle width = " + this.\_width.ToString() + "\nRectangle height = " + this.height.ToString() + "\nRectangle area = " + Area().ToString() + "\n";

}

public void Print()

{

System.Console.WriteLine(ToString());

}

}

class Square : Rectangle, IPrint

{

public Square(double s) : base(s, s) { } //call of rect constructor in square constructor

public override string ToString()

{

return "Square side = " + this.width.ToString() + "\nSquare area = " + Area().ToString() + "\n";

}

public void Print()

{

System.Console.WriteLine(ToString());

}

}

class Circle : GeomFigure, IPrint

{

public double r { get; set; }

public Circle(double r) { this.r = r; }

public override double Area()

{

return 2 \* Math.PI \* r \* r;

}

public override string ToString()

{

return "Circle radius = " + r.ToString() + "\nCircle area = " + Area().ToString() + "\n";

}

public void Print()

{

System.Console.WriteLine(ToString());

}

}

class Program

{

static int menu()

{

int num = 0;

Console.WriteLine("Choose figure");

Console.WriteLine("1 - Rectangle\n" + "2 - Square\n" + "3 - Circle\n" + "4 - Close\n");

num = int.Parse(Console.ReadLine());

return num;

}

static bool check(double val)

{

bool res = true;

if (val < 0)

{

res = false;

}

return res;

}

static int Main(string[] args)

{

double a = 0, b = 0;

while (true)

{

switch (menu())

{

case 1:

{

Console.WriteLine("Enter width: ");

a = int.Parse(Console.ReadLine());

Console.WriteLine("Enter height: ");

b = int.Parse(Console.ReadLine());

check(a);

check(b);

if (check(a) == false || check(b) == false)

{

Console.WriteLine("Sides can't be negative");

break;

}

Rectangle rect = new Rectangle(a, b);

rect.Print();

break;

}

case 2:

{

Console.WriteLine("Enter size of the side: ");

a = int.Parse(Console.ReadLine());

check(a);

if (check(a) == false)

{

Console.WriteLine("Sides can't be negative");

break;

}

Square sq = new Square(a);

sq.Print();

break;

}

case 3:

{

Console.WriteLine("Enter radius: ");

a = int.Parse(Console.ReadLine());

check(a);

if (check(a) == false)

{

Console.WriteLine("Radius can't be negative");

break;

}

Circle circ = new Circle(a);

circ.Print();

break;

}

default:

{

Console.WriteLine("Enter number from 1 to 4");

break;

}

case 4:

{

return 0;

}

}

Console.ReadLine();

Console.Clear();

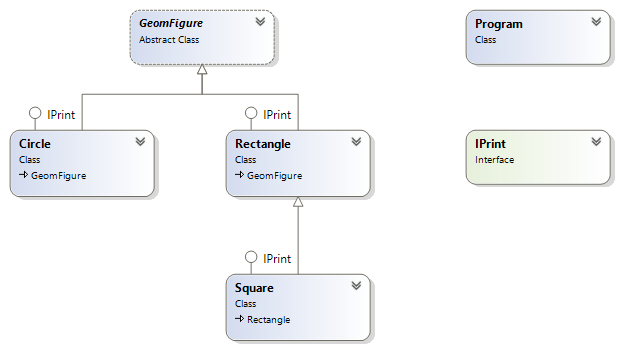
}

}

}

}

**Диаграмма классов**



**Результаты работы**

