Московский государственный технический университет имени Н.Э. Баумана (МГТУ им. Н.Э. Баумана)

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

ПО ДИСЦИПЛИНЕ «БАЗОВЫЕ КОМПОНЕНТЫ ИНТЕРНЕТ ТЕХНОЛОГИЙ»

Выполнил(а): Тимаков А.Г.

Проверил(а): Гапанюк Ю.Е.

Москва, 2017

**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 программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab2

{

abstract class Figure

{

public abstract double area();

}

interface IPrint

{

void Print();

}

class Rectangle : Figure, IPrint

{

private double \_width;

public double width

{

get

{

return \_width;

}

set

{

\_width = value;

}

}

public double height { get; set; }

public Rectangle(double a, double b) { \_width = a; height = b; }

public override double area()

{

return \_width\*height;

}

public override string ToString()

{

return "Rectangle \nWidth = " + this.\_width.ToString() + "\nHeight = " + this.height.ToString() + "\nArea = " + area().ToString() + "\n";

}

public void Print()

{

System.Console.WriteLine(ToString());

}

}

class Square : Rectangle, IPrint

{

public Square(double a) : base(a, a) { }

public override string ToString()

{

return "Square \nSide = " + this.width.ToString() + "\nArea = " + area().ToString() + "\n";

}

public void Print()

{

System.Console.WriteLine(ToString());

}

}

class Circle : Figure, IPrint

{

public double rad { get; set; }

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

public override double area()

{

return Math.PI \* rad \* rad;

}

public override string ToString()

{

return "Circle \nRadius = " + this.rad.ToString() + "\nArea = " + area().ToString() + "\n";

}

public void Print()

{

System.Console.WriteLine(ToString());

}

}

class Program

{

static int menu()

{

int n = 0;

Console.WriteLine("Выберите тип фигуры");

Console.WriteLine("1 - Прямоугольник\n2 - Квадрат\n3 - Круг\n4 - Выход\n");

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

return n;

}

static int Main(string[] args)

{

double a, b;

while (true)

{

switch (menu())

{

case 1:

{

Console.WriteLine("Введите ширину: ");

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

Console.WriteLine("Введите высоту: ");

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

if (a < 0 || b < 0)

{

Console.WriteLine("Значения не могут быть отрицательными");

break;

}

Rectangle rect = new Rectangle(a, b);

rect.Print();

break;

}

case 2:

{

Console.WriteLine("Введите сторону квадрата: ");

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

if (a < 0)

{

Console.WriteLine("Значения не могут быть отрицательными");

break;

}

Square sq = new Square(a);

sq.Print();

break;

}

case 3:

{

Console.WriteLine("Введите радиус: ");

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

if (a < 0)

{

Console.WriteLine("Значения не могут быть отрицательными");

break;

}

Circle c = new Circle(a);

c.Print();

break;

}

case 4:

{

return 0;

}

}

}

}

}

}