**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №2

«Классы. Интерфейсы. Переопределение методов.»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-31 |  | преподаватель каф. ИУ5 |
| Агличеев Михаил |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

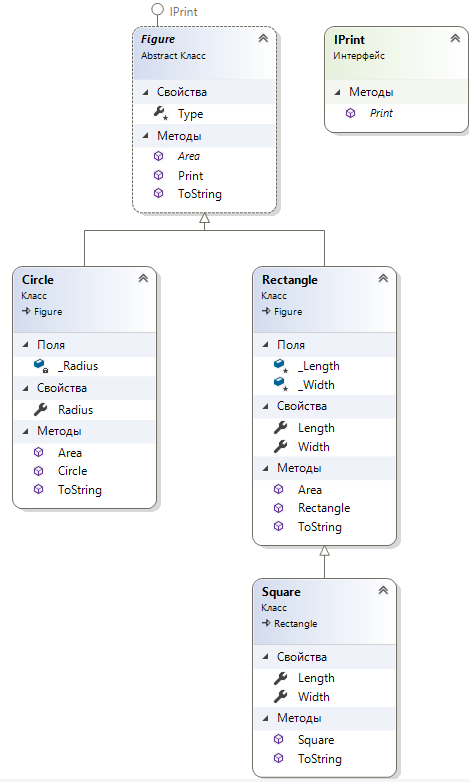
Москва, 2020г.

1. Описание задания

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 «Геометрическая фигура». Ширина и высота объявляются как свойства (property). Класс должен содержать конструктор по параметрам «ширина» и «высота».
4. Класс «Квадрат» наследуется от «Прямоугольник». Класс должен содержать конструктор по длине стороны.
5. Класс «Круг» наследуется от «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().

2. UML-диаграмма классов



3. Листинг программы

using System;

using static System.Math;

namespace Лабораторная\_\_\_2

{

abstract class Figure: IPrint

{

protected string Type { get; set; }

public abstract double Area();

public override string ToString()

{

return this.Type + " площадью " + this.Area();

}

public void Print()

{

Console.WriteLine(this.ToString());

}

}

interface IPrint

{

void Print();

}

class Rectangle: Figure

{

protected double \_Length;

public virtual double Length

{

get { return \_Length; }

set {

if (value >= 0)

{

\_Length = value;

}

else

{

throw new Exception("Длина не может быть отрицательной");

}

}

}

protected double \_Width;

public virtual double Width

{

get { return \_Width; }

set

{

if (value >= 0)

{

\_Width = value;

}

else

{

throw new Exception("Ширина не может быть отрицательной");

}

}

}

public Rectangle(double Length = 0, double Width = 0) {

this.Length = Length;

this.Width = Width;

}

public override double Area()

{

return this.Length \* this.Width;

}

public override string ToString()

{

return $"Прямоугольник со сторонами {this.Length} и {this.Width} и площадью {this.Area()}";

}

}

class Square : Rectangle

{

public Square(double side = 0) : base(side, side) {; }

//Ask how to ban changing length.

public override double Length

{

get { return \_Length; }

set

{

if (value >= 0)

{

\_Width = value;

\_Length = value;

}

else

{

throw new Exception("Сторона не может быть отрицательной");

}

}

}

public override double Width

{

get { return \_Length; }

set

{

if (value >= 0)

{

\_Width = value;

\_Length = value;

}

else

{

throw new Exception("Сторона не может быть отрицательной");

}

}

}

public override string ToString()

{

return $"Квадрат со стороной {this.Length} и площадью {this.Area()}";

}

}

class Circle: Figure

{

private double \_Radius;

public double Radius

{

get { return \_Radius; }

set

{

if (value >= 0)

{

\_Radius = value;

}

else

{

throw new Exception("Радиус не может быть отрицательным");

}

}

}

public Circle(double Radius = 0) {

this.Radius = Radius;

}

public override double Area()

{

return PI \* Pow(Radius, 2);

}

public override string ToString()

{

return $"Круг с радиусом {this.Radius} и площадью {this.Area()}";

}

}

class Program

{

static void Main(string[] args)

{

Rectangle rect = new Rectangle(2.4, 67);

Square square = new Square(5.7);

Circle circle = new Circle(PI);

rect.Print();

square.Print();

circle.Print();

Console.ReadKey();

}

}

}

4. Результаты

