**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №1

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-33Б |  | преподаватель каф. ИУ5 |
| Ветошкин Артём |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2019 г.

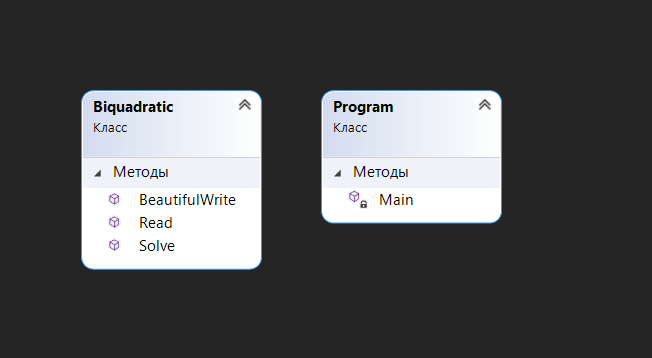
**Постановка задачи**

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 «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.

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().

**Диаграмма классов**



**Текст программы**

Файл Program.cs:

using System;

using System.Collections.Generic;

namespace lab {

class Biquadratic {

public static List<double> Solve((double a, double b, double c) coef) {

if (coef.a is 0) {

if ((-coef.c) \* coef.b >= 0 && coef.b != 0) {

double ans = Math.Sqrt(-coef.c / coef.b);

return ans is 0 ? new List<double> { ans } :

new List<double> { ans, -ans };

}

if (coef.b is 0 && coef.c is 0) {

return new List<double> { double.NaN };

}

return new List<double> {};

}

double D = coef.b \* coef.b - 4 \* coef.a \* coef.c;

if (D < 0) {

return new List<double> { };

}

List<double> roots = default;

double res1 = (-coef.b + Math.Sqrt(D)) / (2.0 \* coef.a);

if (res1 >= 0) {

double ans = Math.Sqrt(res1);

roots = ans is 0 ? new List<double> { ans } :

new List<double> { ans, -ans };

}

else {

roots = new List<double> { };

}

if (D is 0) {

return roots;

}

double res2 = (-coef.b - Math.Sqrt(D)) / (2.0 \* coef.a);

if (res2 >= 0) {

double ans = Math.Sqrt(res2);

if (ans is 0) {

roots.Add(ans);

}

else {

roots.AddRange(new double[] { ans, -ans });

}

}

return roots;

}

public static (double a, double b, double c) Read() {

Console.WriteLine("Please, Enter coefficients of biquadratic by spaces");

string line = Console.ReadLine();

var split\_line = line.Split(' ');

while(split\_line.Length != 3) {

Console.WriteLine("Your input is incorrect, try again."

+ " Example: \"1,0 2,0 3,0\"");

line = Console.ReadLine();

split\_line = line.Split(' ');

}

while (!double.TryParse(split\_line[0], out double none)) {

Console.Write("First coefficient is incorrect, try again."

+ " Example: \"1,0\" \na = ");

line = Console.ReadLine();

split\_line[0] = line.Split(' ')[0];

}

double a = double.Parse(split\_line[0]);

while (!double.TryParse(split\_line[1], out double none)) {

Console.Write("Second coefficient is incorrect, try again."

+ " Example: \"1,0\" \nb = ");

line = Console.ReadLine();

split\_line[1] = line.Split(' ')[0];

}

double b = double.Parse(split\_line[1]);

while (!double.TryParse(split\_line[2], out double none)) {

Console.Write("Third coefficient is incorrect, try again."

+ " Example: \"1,0\" \nc = ");

line = Console.ReadLine();

split\_line[2] = line.Split(' ')[0];

}

double c = double.Parse(split\_line[2]);

return (a, b, c);

}

public static void BeautifulWrite(List<double> roots) {

if(roots.Count == 0) {

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("This biquadratic can't be solved");

Console.ResetColor();

return;

}

Console.ForegroundColor = ConsoleColor.Green;

if (double.IsNaN(roots[0])) {

Console.WriteLine("Biquadratic has infinity solves");

}

else {

for (int i = 1; i <= roots.Count; ++i) {

Console.WriteLine("x" + i.ToString() + " = "

+ roots[i - 1].ToString());

}

}

Console.ResetColor();

}

}

class Program {

static void Main(string[] args) {

Console.WriteLine("Ветошкин Артём, ИУ5-33Б");

Console.WriteLine("Hello, This program solves biquadratic\n");

(double a, double b, double c) coefficient = default;

if (args.Length == 3) {

if (!double.TryParse(args[0], out coefficient.a)

|| !double.TryParse(args[1], out coefficient.b)

|| !double.TryParse(args[2], out coefficient.c)) {

Console.WriteLine("Sorry, coefficients from arg of comand line is incorect.");

coefficient = Biquadratic.Read();

}

}

else {

coefficient = Biquadratic.Read();

}

var roots = Biquadratic.Solve(coefficient);

Biquadratic.BeautifulWrite(roots);

Console.WriteLine("\nIt was a good work, bye!");

}

}

}

**Анализ результатов**

