**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

# Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №1

# «Основные конструкции языка Python»

Вариант 9

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-35Б |  |  |
| Крыжный К.С. |  |  |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2024 г

**Описание задания**

Разработать программу для решения биквадратного уравнения.

1. Программа должна быть разработана в виде консольного приложения на языке Python.
2. Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).
3. Коэффициенты А, В, С могут быть заданы в виде параметров командной строки (вариант задания параметров приведен в конце файла с примером кода). Если они не заданы, то вводятся с клавиатуры в соответствии с пунктом 2. Описание работы с параметрами командной строки.
4. Если коэффициент А, В, С введен или задан в командной строке некорректно, то необходимо проигнорировать некорректное значение и вводить коэффициент повторно пока коэффициент не будет введен корректно. Корректно заданный коэффициент — это коэффициент, значение которого может быть без ошибок преобразовано в действительное число.
5. Дополнительное задание 1 (\*). Разработайте две программы на языке Python - одну с применением процедурной парадигмы, а другую с применением объектно-ориентированной парадигмы.
6. Дополнительное задание 2 (\*). Разработайте две программы - одну на языке Python, а другую на любом другом языке программирования (кроме С++).

**Текст программы**

Python:

import sys

class bisquare():

def \_\_init\_\_(self):

self.coef\_a=0.0

self.coef\_b=0.0

self.coef\_c=0.0

self.kol\_roots=0

self.roots=[]

def input\_coef(self, index, text):

try:

coef\_str=sys.argv[index]

except:

print(text)

coef\_str=input()

coef=float(coef\_str)

return coef

def get\_coef(self):

self.coef\_a=self.input\_coef(1,"Введите A:")

self.coef\_b=self.input\_coef(2,"Введите B:")

self.coef\_c=self.input\_coef(3,"Введите C:")

def solution(self):

a=self.coef\_a

b=self.coef\_b

c=self.coef\_c

d=b\*b-4\*a\*c

if d<0.0:

self.kol\_roots=0

elif d==0.0:

t=(-b+(d\*\*0.5))/(2\*a)

if t<0.0:

self.kol\_roots=0

elif t==0.0:

self.roots.append(t)

self.kol\_roots=1

else:

self.roots.append(t\*\*0.5)

self.roots.append(-(t\*\*0.5))

self.kol\_roots=2

else:

t1=(-b+(d\*\*0.5))/(2\*a)

t2=(-b-(d\*\*0.5))/(2\*a)

if t1<0.0:

if t2<0.0:

self.kol\_roots=0

elif t2==0.0:

self.kol\_roots=1

self.roots.append(0)

else:

self.kol\_roots=2

self.roots.append(t2\*\*0.5)

self.roots.append(-(t2\*\*0.5))

elif t1==0.0:

if t2<0.0:

self.kol\_roots=1

self.roots.append(0)

elif t2==0.0:

self.kol\_roots=1

self.roots.append(0)

else:

self.kol\_roots=3

self.roots.append(0)

self.roots.append(t2\*\*0.5)

self.roots.append(-(t2\*\*0.5))

else:

if t2<0.0:

self.kol\_roots=2

self.roots.append(t1\*\*0.5)

self.roots.append(-(t1\*\*0.5))

elif t2==0.0:

self.kol\_roots=3

self.roots.append(t1\*\*0.5)

self.roots.append(-(t1\*\*0.5))

self.roots.append(0)

else:

self.kol\_roots=4

self.roots.append(t1\*\*0.5)

self.roots.append(-(t1\*\*0.5))

self.roots.append(t2\*\*0.5)

self.roots.append(-(t2\*\*0.5))

def print\_roots(self):

if self.kol\_roots != len(self.roots):

print(('Ошибка. Уравнение содержит {} действительных корней, ' + \

'но было вычислено {} корней.').format(self.kol\_roots, len(self.kol\_roots)))

else:

if self.kol\_roots == 0:

print('Нет корней')

elif self.kol\_roots == 1:

print('Один корень: {}'.format(self.roots[0]))

elif self.kol\_roots == 2:

print('Два корня: {} и {}'.format(self.roots[0], \

self.roots[1]))

elif self.kol\_roots == 3:

print('Три корня: {}, {} и {}'.format(self.roots[0], \

self.roots[1], \

self.roots[2]))

elif self.kol\_roots == 4:

print('Четыре корня: {}, {}, {} и {}'.format(self.roots[0], \

self.roots[1], \

self.roots[2], \

self.roots[3]))

def main():

k = bisquare()

k.get\_coef()

k.solution()

k.print\_roots()

if \_\_name\_\_ == "\_\_main\_\_":

main()

Python (доп задание):

import sys

def input\_coef(index, text):

try:

coef\_str = sys.argv[index]

except:

print(text)

coef\_str = input()

coef = float(coef\_str)

return coef

def get\_coef():

coef\_a = input\_coef(1, "Введите A:")

coef\_b = input\_coef(2, "Введите B:")

coef\_c = input\_coef(3, "Введите C:")

return coef\_a, coef\_b, coef\_c

def solution(a, b, c):

roots = []

d = b \* b - 4 \* a \* c

if d < 0.0:

return 0, roots

elif d == 0.0:

t = (-b + (d\*\*0.5)) / (2 \* a)

if t < 0.0:

return 0, roots

elif t == 0.0:

roots.append(t)

return 1, roots

else:

roots.append(t\*\*0.5)

roots.append(-(t\*\*0.5))

return 2, roots

else:

t1 = (-b + (d\*\*0.5)) / (2 \* a)

t2 = (-b - (d\*\*0.5)) / (2 \* a)

if t1 < 0.0:

if t2 < 0.0:

return 0, roots

elif t2 == 0.0:

roots.append(0)

return 1, roots

else:

roots.append(t2\*\*0.5)

roots.append(-(t2\*\*0.5))

return 2, roots

elif t1 == 0.0:

if t2 < 0.0:

roots.append(0)

return 1, roots

elif t2 == 0.0:

roots.append(0)

return 1, roots

else:

roots.append(0)

roots.append(t2\*\*0.5)

roots.append(-(t2\*\*0.5))

return 3, roots

else:

if t2 < 0.0:

roots.append(t1\*\*0.5)

roots.append(-(t1\*\*0.5))

return 2, roots

elif t2 == 0.0:

roots.append(t1\*\*0.5)

roots.append(-(t1\*\*0.5))

roots.append(0)

return 3, roots

else:

roots.append(t1\*\*0.5)

roots.append(-(t1\*\*0.5))

roots.append(t2\*\*0.5)

roots.append(-(t2\*\*0.5))

return 4, roots

def print\_roots(kol\_roots, roots):

if kol\_roots != len(roots):

print(f'Ошибка. Уравнение содержит {kol\_roots} действительных корней, ' +

f'но было вычислено {len(roots)} корней.')

else:

if kol\_roots == 0:

print('Нет корней')

elif kol\_roots == 1:

print(f'Один корень: {roots[0]}')

elif kol\_roots == 2:

print(f'Два корня: {roots[0]} и {roots[1]}')

elif kol\_roots == 3:

print(f'Три корня: {roots[0]}, {roots[1]} и {roots[2]}')

elif kol\_roots == 4:

print(f'Четыре корня: {roots[0]}, {roots[1]}, {roots[2]} и {roots[3]}')

def main():

coef\_a, coef\_b, coef\_c = get\_coef()

kol\_roots, roots = solution(coef\_a, coef\_b, coef\_c)

print\_roots(kol\_roots, roots)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Java:

public class Bisquare {

private double coefA;

private double coefB;

private double coefC;

private int kolRoots;

private double[] roots;

// Конструктор

public Bisquare() {

this.coefA = 0.0;

this.coefB = 0.0;

this.coefC = 0.0;

this.kolRoots = 0;

this.roots = new double[0]; // Инициализация пустого массива

}

// Геттеры и сеттеры

public double getCoefA() {

return coefA;

}

public void setCoefA(double coefA) {

this.coefA = coefA;

}

public double getCoefB() {

return coefB;

}

public void setCoefB(double coefB) {

this.coefB = coefB;

}

public double getCoefC() {

return coefC;

}

public void setCoefC(double coefC) {

this.coefC = coefC;

}

public int getKolRoots() {

return kolRoots;

}

public void setKolRoots(int kolRoots) {

this.kolRoots = kolRoots;

}

public double[] getRoots() {

return roots;

}

public void setRoots(double[] roots) {

this.roots = roots;

}

// Ввод коэффициентов с консоли или командной строки

public double inputCoef(String[] args, int index, String text) {

Scanner scanner = new Scanner(System.in);

String coefStr = "";

try {

coefStr = args[index];

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println(text);

coefStr = scanner.nextLine();

}

return Double.parseDouble(coefStr);

}

public void getCoef(String[] args) {

this.setCoefA(inputCoef(args, 0, "Введите A:"));

this.setCoefB(inputCoef(args, 1, "Введите B:"));

this.setCoefC(inputCoef(args, 2, "Введите C:"));

}

// Решение уравнения

public void solution() {

double a = this.getCoefA();

double b = this.getCoefB();

double c = this.getCoefC();

double d = b \* b - 4 \* a \* c; // Дискриминант

if (d < 0.0) {

this.setKolRoots(0);

} else if (d == 0.0) {

double t = (-b + Math.sqrt(d)) / (2 \* a);

if (t < 0.0) {

this.setKolRoots(0);

} else if (t == 0.0) {

this.setRoots(new double[1]);

this.getRoots()[0] = t;

this.setKolRoots(1);

} else {

this.setRoots(new double[2]);

this.getRoots()[0] = Math.sqrt(t);

this.getRoots()[1] = -Math.sqrt(t);

this.setKolRoots(2);

}

} else {

double t1 = (-b + Math.sqrt(d)) / (2 \* a);

double t2 = (-b - Math.sqrt(d)) / (2 \* a);

if (t1 < 0.0) {

if (t2 < 0.0) {

this.setKolRoots(0);

} else if (t2 == 0.0) {

this.setRoots(new double[1]);

this.getRoots()[0] = 0;

this.setKolRoots(1);

} else {

this.setRoots(new double[2]);

this.getRoots()[0] = Math.sqrt(t2);

this.getRoots()[1] = -Math.sqrt(t2);

this.setKolRoots(2);

}

} else if (t1 == 0.0) {

if (t2 < 0.0) {

this.setRoots(new double[1]);

this.getRoots()[0] = 0;

this.setKolRoots(1);

} else if (t2 == 0.0) {

this.setRoots(new double[1]);

this.getRoots()[0] = 0;

this.setKolRoots(1);

} else {

this.setRoots(new double[3]);

this.getRoots()[0] = 0;

this.getRoots()[1] = Math.sqrt(t2);

this.getRoots()[2] = -Math.sqrt(t2);

this.setKolRoots(3);

}

} else {

if (t2 < 0.0) {

this.setRoots(new double[2]);

this.getRoots()[0] = Math.sqrt(t1);

this.getRoots()[1] = -Math.sqrt(t1);

this.setKolRoots(2);

} else if (t2 == 0.0) {

this.setRoots(new double[3]);

this.getRoots()[0] = Math.sqrt(t1);

this.getRoots()[1] = -Math.sqrt(t1);

this.getRoots()[2] = 0;

this.setKolRoots(3);

} else {

this.setRoots(new double[4]);

this.getRoots()[0] = Math.sqrt(t1);

this.getRoots()[1] = -Math.sqrt(t1);

this.getRoots()[2] = Math.sqrt(t2);

this.getRoots()[3] = -Math.sqrt(t2);

this.setKolRoots(4);

}

}

}

}

// Печать корней

public void printRoots() {

if (this.getKolRoots() != this.getRoots().length) {

System.out.println(String.format("Ошибка. Уравнение содержит %d действительных корней, но было вычислено %d корней.",

this.getKolRoots(), this.getRoots().length));

} else {

switch (this.getKolRoots()) {

case 0:

System.out.println("Нет корней");

break;

case 1:

System.out.println("Один корень: " + this.getRoots()[0]);

break;

case 2:

System.out.println(String.format("Два корня: %f и %f", this.getRoots()[0], this.getRoots()[1]));

break;