**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»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-31Б |  | преподаватель каф. ИУ5 |
| Маркин Денис |  | Нардид Анатолий Николаевич |
|  |  |  |
|  |  |  |

Москва, 2024 г.

Постановка задачи

Задание.

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

Lab1NoPar.py

import sys

import math

#Без парадигмы

def get\_coef(to\_print, i):

    try:

        coef\_s= sys.argv[i]

        coef = float(coef\_s)

    except (IndexError, ValueError):

        print(to\_print)

        while(True):

            coef\_s = input()

            try:

                coef = float(coef\_s)

                break

            except ValueError:

                print("Неверный ввод! Введите число:")

    return coef

def get\_roots(a,b,c):

    roots = []

    D = b\*\*2-4\*a\*c

    if(D <0):

        return roots

    elif(D == 0):

        x = -b/2/a

        if(x>0):

            roots += [round(x\*\*0.5, 4), round(-x\*\*0.5, 4)]

    else:

        x1 = -(b+D\*\*0.5)/2/a

        x2 = -(b-D\*\*0.5)/2/a

        if(x1>0):

            roots += [round(-x1\*\*0.5, 4), round(x1\*\*0.5, 4)]

        if(x2>0):

            roots += [round(-x2\*\*0.5, 4), round(x2\*\*0.5, 4)]

    return roots

def main():

    a = get\_coef("Введите коэффициент a:", 1)

    b = get\_coef("Введите коэффициент b:", 2)

    c = get\_coef("Введите коэффициент c:", 3)

    roots = get\_roots(a, b, c)

    if len(roots) == 0:

        print(f"Уравнение {int(a)}x^4 + {int(b)}x^2 + {int(c)}= 0 не имеет корней.")

    elif len(roots) <= 4:

        root\_list = ', '.join(map(str, roots))

        print(f"Уравнение {int(a)}x^4 + {int(b)}x^2 + {int(c)} = 0 имеет корни: {root\_list}.")

    else:

        print("Уравнение имеет больше 4-х корней, ОШИБКА!")

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

    main()

Lab1FuncPar.py

import sys

#Процедурная парадигма

def input\_coef():

    try:

        a\_s = sys.argv[1]

        b\_s =sys.argv[2]

        c\_s = sys.argv[3]

        a= float(a\_s)

        b= float(b\_s)

        c = float(c\_s)

    except (IndexError, ValueError):

        a = float(vvod("a"))

        b = float(vvod("b"))

        c = float(vvod("c"))

    return a, b, c

def vvod(a):

    print("Введите число " +a)

    while(True):

            coef\_s = input()

            try:

                coef = float(coef\_s)

                break

            except ValueError:

                print("Неверный ввод! Введите число еще раз:")

    return coef

def calc\_disc(a, b, c):

    return b\*\*2 - 4\*a\*c

def find\_roots(a, b, c):

    D = calc\_disc(a, b, c)

    roots = []

    if(D<0):

        return None

    elif(D == 0):

        x = -b/2/a

        if(x>0):

            roots+=[round(x\*\*0.5, 4), round(-x\*\*0.5), 4]

    else:

        x1 = -(b+D\*\*0.5)/2/a

        x2 = -(b-D\*\*0.5)/2/a

        if(x1>0):

            roots += [round(-x1\*\*0.5, 4), round(x1\*\*0.5, 4)]

        if(x2>0):

            roots += [round(-x2\*\*0.5, 4), round(x2\*\*0.5, 4)]

    return roots

def print\_roots(roots):

    if roots is None:

        print("Уравнение не имеет действительных корней.")

    elif len(roots) == 2:

        print(f"Уравнение имеет два корня: {roots[0]} и {roots[1]}")

    else:

        print(f"Уравнение имеет четыре корня: +-{roots[1]} и +- {roots[3]}")

def main():

    try:

        a, b, c = input\_coef()

        roots = find\_roots(a, b, c)

        print\_roots(roots)

    except ValueError as e:

        print(f"Ошибка ввода: {e}")

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

    main()

Lab1OOPPar.py

#Объектно-ориентированная парадигма

class Bikvadrat:

    def \_\_init\_\_(self, a, b, c):

        self.a = a

        self.b = b

        self.c = c

    def solve(self):

        D = self.b\*\*2 - 4 \* self.a \* self.c

        if D< 0:

            return "Уравнение не имеет корней"

        y1 = (-self.b + D\*\*0.5) / (2 \* self.a)

        y2 = (-self.b - D\*\*0.5) / (2 \* self.a)

        roots = []

        if y1 >= 0:

            roots.append(round(y1\*\*0.5,4))

            roots.append(round(-y1\*\*0.5, 4))

        if y2 >= 0:

            roots.append(round(y2\*\*0.5,4))

            roots.append(round(-y2\*\*0.5,4))

        return roots if roots else "Уравнение не имеет корней"

def main():

    a = float(input("Введите коэффициент a: "))

    b = float(input("Введите коэффициент b: "))

    c = float(input("Введите коэффициент c: "))

    ans = Bikvadrat(a, b, c)

    roots = ans.solve()

    print("Корни уравнения:", roots)

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

    main()

Lab1.go

// There are no classes

//Программа на другом языке (golang)

package main

import (

    "fmt"

    "math"

)

type Bikvadrat struct {

    a float64

    b float64

    c float64

}

func NewBikvadrat(a float64, b float64, c float64) \*Bikvadrat {

    return &Bikvadrat{a: a, b: b, c: c}

}

func RoundToDecimal(value float64, places int) float64 {

    pow := math.Pow(10, float64(places))

    return math.Round(value\*pow) / pow

}

func Solve(n Bikvadrat) []float64 {

    D := n.b\*n.b - 4\*n.a\*n.c

    roots := []float64{}

    if D < 0 {

        return roots

    } else if D == 0 {

        x := -n.b / 2 / n.a

        if x > 0 {

            roots = append(roots, RoundToDecimal(math.Pow(float64(x), float64(0.5)), 4))

            roots = append(roots, RoundToDecimal(-math.Pow(float64(x), float64(0.5)), 4))

        }

    } else {

        x1 := -(n.b + math.Pow(float64(D), float64(0.5))) / 2 / float64(n.a)

        x2 := -(n.b - math.Pow(float64(D), float64(0.5))) / 2 / float64(n.a)

        if x1 > 0 {

            roots = append(roots, RoundToDecimal(math.Pow(float64(x1), float64(0.5)), 4))

            roots = append(roots, RoundToDecimal(-math.Pow(float64(x1), float64(0.5)), 4))

        }

        if x2 > 0 {

            roots = append(roots, RoundToDecimal(math.Pow(float64(x2), float64(0.5)), 4))

            roots = append(roots, RoundToDecimal(-math.Pow(float64(x2), float64(0.5)), 4))

        }

    }

    return roots

}

func main() {

    var a int16

    fmt.Print("Введите a: ")

    fmt.Scan(&a)

    var b int16

    fmt.Print("Введите b: ")

    fmt.Scan(&b)

    var c int16

    fmt.Print("Введите c: ")

    fmt.Scan(&c)

    Ans := NewBikvadrat(float64(a), float64(b), float64(c))

    roots := []float64{}

    roots = Solve(\*Ans)

    if len(roots) == 0 {

        fmt.Print("Уравнение не имеет корней")

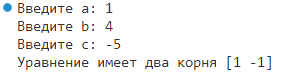
    } else if len(roots) == 2 {

        fmt.Print("Уравнение имеет два корня ", roots)

    } else {

        fmt.Print("Уравнение имеет четыре корня ", roots)

Анализ результатов

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Язык golang

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Процедурная парадигма

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описаниеИзображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Объектно-ориентированная парадигма

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Без парадигмы