**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 4

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-35Б |  |  |
| Бодня В.А. |  |  |
| 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

import math

def get\_coef(index, prompt):

    try:

        coef\_str = sys.argv[index]

    except:

        print(prompt)

        coef\_str = input()

    coef = float(coef\_str)

    return coef

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

    result = []

    D = b\*b - 4\*a\*c

    if D == 0.0:

        root = -b / (2.0\*a)

        result.append(root)

    elif D > 0.0:

        sqD = math.sqrt(D)

        root1 = (-b + sqD) / (2.0\*a)

        root2 = (-b - sqD) / (2.0\*a)

        result.append(root1)

        result.append(root2)

    final\_result = []

    for root in result:

        if root == 0.0:

            final\_result.append(0.0)

        elif root > 0.0:

            final\_result.append(math.sqrt(root))

            final\_result.append(-math.sqrt(root))

    return final\_result

def main():

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

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

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

    roots = get\_roots(a,b,c)

    len\_roots = len(roots)

    if len\_roots == 0:

        print('Нет корней')

    elif len\_roots == 1:

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

    elif len\_roots == 2:

        print('Два корня: {} и {}'.format(roots[0], roots[1]))

    elif len\_roots == 3:

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

    elif len\_roots == 4:

        print('Три корня: {}, {}, {} и {}'.format(roots[0], roots[1], roots[2], roots[3]))

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

    main()

**Java:**

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class Main {

    private static boolean isEqual(double x, double y) {

        return Math.abs(x - y) < 1e-10; // Используем небольшую epsilon для сравнения

    }

    private static List<Double> getRoots(double a, double b, double c) {

        List<Double> result = new ArrayList<>();

        int d = (int) (b \* b - 4 \* a \* c);

        if (isEqual(d, 0)) {

            result.add(-b / (2 \* a));

        } else if (d > 0) {

            double sqD = Math.sqrt(d);

            result.add((-b + sqD) / (2.0 \* a));

            result.add((-b - sqD) / (2.0 \* a));

        }

        List<Double> finalResult = new ArrayList<>();

        for (double root : result) {

            if (isEqual(root, 0.0)) {

                finalResult.add(0.0);

            } else if (root > 0) {

                finalResult.add(Math.sqrt(root));

                finalResult.add(-Math.sqrt(root));

            }

        }

        return finalResult;

    }

    public static void main(String[] args) {

        double a, b, c;

        Scanner scanner = new Scanner(System.in);

        System.out.print("Введите коэффициент A: ");

        a = scanner.nextDouble();

        System.out.print("Введите коэффициент B: ");

        b = scanner.nextDouble();

        System.out.print("Введите коэффициент C: ");

        c = scanner.nextDouble();

        // Проверка на 0 для коэффициента a

        if (isEqual(a, 0)) {

            System.out.println("Коэффициент A не может быть равен 0.");

            return;

        }

        List<Double> roots = getRoots(a, b, c);

        int lenRoots = roots.size();

        switch (lenRoots) {

            case 0:

                System.out.println("Нет корней.");

                break;

            case 1:

                System.out.println("Один корень: " + roots.get(0));

                break;

            case 2:

                System.out.println("Два корня: " + roots.get(0) + " и " + roots.get(1));

                break;

            case 3:

                System.out.println("Три корня: " + roots.get(0) + ", " + roots.get(1) + " и " + roots.get(2));

                break;

            case 4:

                System.out.println("Четыре корня: " + roots.get(0) + ", " + roots.get(1) + ", " + roots.get(2) + ", " + roots.get(3));

                break;

            default:

                break;

        }

        scanner.close();

    }

}

**С++:**

#include <iostream>

#include <vector>

#include <limits>

#include <cmath>

using namespace std;

bool is\_equal(double x, double y) {

    return std::fabs(x - y) < std::numeric\_limits<double>::epsilon();

}

vector<double> get\_roots(double a, double b, double c) {

    vector<double> result;

    int d = b \* b - 4 \* a \* c;

    if (is\_equal(d, 0)) {

        result.push\_back(-b / (2 \* a));

    } else if (d > 0) {

        double sqD = sqrt(d);

        result.push\_back((-b + sqD) / (2.0 \* a));

        result.push\_back((-b - sqD) / (2.0 \* a));

    }

    vector<double> final\_result;

    for (double root : result) {

        if (is\_equal(root, 0.0)) {

            final\_result.push\_back(0);

        } else if (root > 0) {

            final\_result.push\_back(sqrt(root));

            final\_result.push\_back(-sqrt(root));

        }

    }

    return final\_result;

}

int main(int argc, char\* argv[]) {

    double a, b, c;

    if (argc == 4) {

        a = atof(argv[1]);

        b = atof(argv[2]);

        c = atof(argv[3]);

    } else {

        cout << "Введите коэффициент A: ";

        cin >> a;

        cout << "Введите коэффициент B: ";

        cin >> b;

        cout << "Введите коэффициент C: ";

        cin >> c;

    }

    // Проверка на 0 для коэффициента a

    if (is\_equal(a, 0)) {

        cout << "Коэффициент A не может быть равен 0." << endl;

        return -1;

    }

    vector<double> roots = get\_roots(a, b, c);

    int len\_roots = roots.size();

    switch (len\_roots) {

        case 0:

            cout << "Нет корней." << endl;

            break;

        case 1:

            cout << "Один корень: " << roots[0] << endl;

            break;

        case 2:

            cout << "Два корня: " << roots[0] << " и " << roots[1] << endl;

            break;

        case 3:

            cout << "Три корня: " << roots[0] << ", " << roots[1] << " и " << roots[2] << endl;

            break;

        case 4:

            cout << "Четыре корня: " << roots[0] << ", " << roots[1] << ", " << roots[2] << ", " << roots[3] << endl;

            break;

        default:

            break;

    }

    return 0;

}

**Экранные формы с примерами выполнения программы**

