**Разработка на языке программирования Rust.**

Выполнил: Беликов Константин  
Группа: ИУ5-36Б  
Дата: 17.09.24г.

Описание задания:

1. Разработать программу для решения биквадратного уравнения.
2. Программа должна быть разработана в виде консольного приложения на языке Python.
3. Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).
4. 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.
5. 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.
6. Дополнительное задание 1 (\*). Разработайте две программы на языке Python - одну с применением процедурной парадигмы, а другую с применением объектно-ориентированной парадигмы.
7. Дополнительное задание 2 (\*). Разработайте две программы - одну на языке Python, а другую на любом другом языке программирования (кроме С++).

Код программы:

use std::io;

//Функция для получения числа с клавиатуры

fn take() -> f64 {

    let mut input = String::new();

    io::stdin().read\_line(&mut input).unwrap();

    return input.trim().parse().unwrap();

}

//Функция записывает найденные корни в массив и возвращает количество корней

fn get\_roots(a: f64, b: f64, c: f64, roots: &mut [f64]) -> i32 {

    let D: f64 = b.powi(2) - 4.0 \* a \* c;

    if D < 0.0 {return 0;

    } else if D == 0.0 {

        roots[0] = -1.0 \* b / (2.0 \* a);

        return 1;

    } else {

        roots[0] = (-1.0 \* b - D.powf(0.5)) / (2.0 \* a);

        roots[1] = (-1.0 \* b + D.powf(0.5)) / (2.0 \* a);

        return 2;

    };

}

fn main() {

    let a = take();

    let b = take();

    let c = take();

    let mut roots: [f64; 2] = [0.0; 2];

    //Отображение результата

    match get\_roots(a, b, c, &mut roots) {

        2 => println!("{}, {}", roots[0], roots[1]),

        1 => println!("{}", roots[0]),

        \_ => println!("no roots"),

    }

}

Снимки экрана:  
