**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 №3

«Изучение основных конструкций языка Rust. Тесты.»

Выполнил:

студент группы ИУ5Ц-52Б

Мефодьев Илья

Проверил:

Нардид А.Н.

2024 г.

**Задание**

Разработать программу для решения биквадратного уравнения

1. Программа должна быть разработана на языке Rust.

2. Коэффициенты А, В, С задаются в виде параметров командной строки. Вычисляется дискриминант и корни уравнения (в зависимости от дискриминанта).

3. Биквадратное уравнение вычисляется через функцию, возвращающую строку с результатом вычисления.

4. Написать тесты (минимум 3 штуки).

**Текст программы**

use std::env;

*// взято задание из лабы 1 и подогнано под тесты.*

fn find\_root(A: i32, B:i32, C:i32) -> String{

    let D = B \* B - 4 \* A \* C;

*//let D = findD!(A, B, C);*

    let *mut* result = String::new();

    if (D < 0){

        result = "Корней нет!".to\_string();

    }

    else if (D == 0){

        let answer= -B / (2 \* A);

        let answer = answer as f64;

        if (answer < 0.0){

            result = "Корней нет!".to\_string();

        }

        else{

*//println!("x = {0}", answer.sqrt().round());*

            result = "x = ".to\_string() + &answer.sqrt().round().to\_string();

        }

    }

    else if (D > 0){

        let D = D as f64;

        let B = B as f64;

        let A = A as f64;

        let answer1 = (-B - D.sqrt()) / (2.0 \* A);

        let answer2 = (-B + D.sqrt()) / (2.0 \* A);

        if (answer1 < 0.0 && answer2 < 0.0){ *//квадрат не может быть отрицательным!*

            result = "Корней нет!".to\_string();

        }

        if (answer1 >= 0.0){

            result = "x1 = ".to\_string() + &answer1.sqrt().round().to\_string();

*//println!("x1 = {0}", answer1.sqrt().round());*

        }

        if (answer2 >= 0.0){

            if(!result.is\_empty()){result.push\_str("; ");}*// разделитель*

*//println!("x2 = {0}", answer2.sqrt().round());*

            let preres = "x2 = ".to\_string() + &answer2.sqrt().round().to\_string();

            result.push\_str(&preres);

        }

    }

    result

}

#[cfg(test)]

mod tests{ *// сами тесты.*

    use std::result;

    use super::\*;

    #[test]

    fn test\_1(){

        let result = find\_root(4,-5,1);

        assert\_eq!(result, "x1 = 1; x2 = 1");

    }

    #[test]

    fn test\_2(){

        let result = find\_root(7,5,3);

        assert\_eq!(result, "Корней нет!");

    }

    #[test]

    fn test\_3(){

        let result = find\_root(1,-2,1);

        assert\_eq!(result, "x = 1");

    }

}

fn main(){

    let args: Vec<String> = env::args().collect();

    if(args.len() != 4){

        println!("Ошибка, неверное число параметров!");

        return

    }

    let A = args[1].parse::<i32>().unwrap(); *// unwrap снимает try catch exeption - для упрощения*

    let B = args[2].parse::<i32>().unwrap();

    let C = args[3].parse::<i32>().unwrap();

    println!("Введено уравнение {A}x^4 + {B}x^2 + {C} = 0");

    let res = find\_root(A, B, C);

    println!("{res}");

}

**Скриншоты с примерами выполнения программы**

