|  |  |  |
| --- | --- | --- |
| Защищено:  Нардид А.Н.    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |  | Демонстрация:  Нардид А.Н.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**Отчет по ДЗ по курсу**

**Парадигмы и конструкции языков программирования**

**ГУИМЦ**

#### Тема работы: " Разработка на языке программирования Rust. "

4

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-51Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Омарова З.Р. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Москва, МГТУ - 2024 г.

# Тема и задание для выполнения лабораторной работы.

# Цель лабораторной работы: изучение возможностей языка программирования Rust.

# Реализуйте любое из заданий курса на языке программирования Rust.

# Разработайте хотя бы один макрос.

# Разработайте модульные тесты (не менее 3 тестов).

# Листинг программы

**lab\src\main.rs**

use std::f64;

fn solve\_biquadratic(a: f64, b: f64, c: f64) -> Vec<f64> {

    let mut solutions = Vec::new();

    // Решаем квадратное уравнение относительно y

    let discriminant = b \* b - 4.0 \* a \* c;

    if discriminant < 0.0 {

        return solutions; // Нет действительных решений

    }

    // Находим корни уравнения для y

    let y1 = (-b + discriminant.sqrt()) / (2.0 \* a);

    let y2 = (-b - discriminant.sqrt()) / (2.0 \* a);

    // Находим x, если y >= 0

    if y1 >= 0.0 {

        solutions.push(y1.sqrt());

        solutions.push(-y1.sqrt());

    }

    if y2 >= 0.0 && y1 != y2 {

        solutions.push(y2.sqrt());

        solutions.push(-y2.sqrt());

    }

    solutions

}

//квадратное уравнение вида: ax^4+bx^2+c=0

fn main() {

    let a = 1.0;

    let b = -5.0;

    let c = 4.0;

    let solutions = solve\_biquadratic(a, b, c);

    if solutions.is\_empty() {

        println!("Нет действительных решений");

    } else {

        println!("Решения биквадратного уравнения: {:?}", solutions);

    }

}

# lab\_macros\src\main.rs

macro\_rules! func

{

    ($a:expr)=>{

    $a\*$a}

}

fn main() {

   let a=4;

   let sq = func!(a);

    println!("Квадрат {} равен {}",a,sq);

}

# unit\_test\src\main.rs

pub fn add(a: i32, b: i32) -> i32 {

a + b }

fn main() {

let result = add(5, 3);

println!("Результат: {}", result);

}

#[cfg(test)]

mod tests {

use super::\*; // Импортируем функции и структуры из текущего модуля

#[test]

fn test\_add\_positive\_numbers() {

assert\_eq!(add(2, 3), 5); // Проверяем, что сумма 2 и 3 равна 5

}

#[test]

fn test\_add\_negative\_numbers() {

assert\_eq!(add(-2, -3), -5); // Проверяем, что сумма -2 и -3 равна -5

}

#[test]

fn test\_add\_mixed\_numbers() {

assert\_eq!(add(-2, 3), 2); // Проверяем, что сумма -2 и 3 равна 1

}

#[test]

fn test\_add\_zero() {

assert\_eq!(add(0, 5), 5); // Проверяем, что сумма 0 и 5 равна 5

assert\_eq!(add(5, 0), 5); // Проверяем, что сумма 5 и 0 равна 5

}

}

# Результаты работы программы

**lab\src\main.rs**: Решения биквадратного уравнения: [2.0, -2.0, 1.0, -1.0]  
**lab\src\main.rs**: Квадрат 4 равен 16

**unit\_test\src\main.rs:**

Compiling unit\_test v0.1.0 (C:\Users\Admin\Desktop\Rust\unit\_test)

Finished `dev` profile [unoptimized + debuginfo] target(s) in 2.15s

Running `target\debug\unit\_test.exe`

Результат: 8