Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»
Курс «Базовые компоненты интернет-технологий»

Отчёт по лабораторной работе №6
«Разработка на языке программирования Rust»

Выполнил:
студент группы ИУ5-35Б
Солопов Александр

Проверил:
Преподаватель кафедры
ИУ5 Гапанюк Ю.Е.

Москва, 2023 г.

Биквадратное уравнение

Текст программы

```
use std::io;

#[derive(Debug,Clone, Copy)]
enum Roots {

    NoRoots,
    OneRoot(f32),
    TwoRoots{ root1: f32, root2 : f32 }

}

#[derive(Debug,Clone, Copy)]
enum Bi_Roots{
    NoRoots,
    TwoRoots
    {
        root1: f32,
        root2: f32,
    },
    FourRoots{
        root1: f32,
        root2: f32,
        root3: f32,
        root4: f32,
    }
}
#[derive(Debug,Clone, Copy)]
struct Square_Equation {
    a : f32,
    b: f32,
    c: f32,
    D: f32,
    result: Roots

}

impl Square_Equation {
    fn calculate_roots(&mut self)
    {
        self.D=self.b.powf(2.0)-4.0*self.a*self.c;
        self.result  = {
            if self.D < 0.0 {
                Roots::NoRoots

        } else if self.D == 0.0 {
            let root: f32 = (-self.b) / (2.0 * self.a);
            Roots::OneRoot(root)
        }
        else {
            let r1: f32 = (-self.b - self.D.powf(0.5)) / (2.0 * self.a);
            let r2: f32 = (-self.b + self.D.powf(0.5)) / (2.0 * self.a);
            Roots::TwoRoots { root1: r1, root2: r2 }

        }
    }
}
```

```
    fn get_coefs(&mut self)
    {
        self.a=Square_Equation::get_coef("Введите коэффициент a:");
        self.b=Square_Equation::get_coef("Введите коэффициент b:");
        self.c=Square_Equation::get_coef("Введите коэффициент c:");
    }
    fn get_coef(message: &str) -> f32
    {


        return loop {
            let mut res= String::new();

            println!("{}", message);
            io::stdin()
            .read_line(&mut res)
            .expect("Неверно введена строка");
            match res.trim().parse() {
                Ok(val) => {
                    break val
                }
                Err(_)=> {
                    continue;
                }
            }

        }
    }
}


struct Bi_Square_Equation{
    rts: Square_Equation,
    result: Bi_Roots
}

impl Bi_Square_Equation {

    fn get_bi_coefs(&mut self)
    {

            self.result = match self.rts.result{
                Roots::NoRoots => Bi_Roots::NoRoots,
                Roots::OneRoot(rt) => Bi_Roots::TwoRoots { root1: -
rt.powf(0.5), root2: rt.powf(0.5) },
                Roots::TwoRoots { root1, root2 } =>
                    Bi_Roots::FourRoots { root1: -root1.powf(0.5),
root2:root1.powf(0.5), root3:-root2.powf(0.5), root4: root2.powf(0.5)}
            }


    }
}




fn main() {


    let mut eq = Square_Equation {
        a : 0.0,
        b: 0.0,
        c: 0.0,
```

```
        D: 0.0,
        result: Roots::NoRoots
    };

    eq.get_coefs();
    eq.calculate_roots();
    let mut bi_eq: Bi_Square_Equation = Bi_Square_Equation { rts: eq
        , result: Bi_Roots::NoRoots };


    println!("a: {}, b: {}, c: {}",eq.a,eq.b,eq.c);
    println!("Result: {:?}",eq.result);
    let text_res = match eq.result {
        Roots::NoRoots => format!("Корней нет"),
        Roots::OneRoot(rt) => format!("Один корень => {}", rt),
        Roots::TwoRoots { root1, root2 } => format!("Два корня => {} и {}",
root1, root2),
    };
    println!("{}\n\n", text_res);


    println!("Result: {:?}",bi_eq.result);
    let text_res = match bi_eq.result {
        Bi_Roots::NoRoots => format!("Корней нет"),
        Bi_Roots::TwoRoots { root1, root2 } => format!("Два корня => {} и
{}", root1, root2),
        Bi_Roots::FourRoots { root1, root2 ,root3 ,root4 } => format!("4
корня => {} и {}, {} и {} ", root1, root2,root3,root4),
    };
    println!("{}", text_res);


}
```

Результаты работы программы

```
Введите коэффициент a:
1
Введите коэффициент b:
2
Введите коэффициент c:
3
a: 1, b: 2, c: 3
Result: NoRoots
Корней нет
```

```
Введите коэффициент a:
4
Введите коэффициент b:
-5
Введите коэффициент c:
1
a: 4, b: -5, c: 1
Result: TwoRoots { root1: 0.25, root2: 1.0 }
Два корня => 0.25 и 1
```