*Государственное образовательное учреждение высшего профессионального образования*

***«Московский государственный технический университет имени Н.Э. Баумана»***
***(МГТУ им. Н.Э. Баумана)***

---

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Системы обработки информации и управления» (ИУ-5)

# ЛАБОРАТОРНАЯ РАБОТА

## № 1

Основные конструкции языка Python

Группа ИУ5-35Б

Студент _____ **16.12.2024 /А. А. Торопыгин/**

(Подпись, дата) (И.О.Фамилия)

Преподаватель _____ _____ **/Ю. Е. Гапанюк/**

(Подпись, дата) (И.О.Фамилия)

2024

**Задание:**
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 A, B, C, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).
3. Коэффициенты A, B, C 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 A, B, C 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 C++).

## Текст программы (Python):

```python
import sys

def get_coef(index, prompt):
    try:
        coef_str = sys.argv[index]
    except:
        print(prompt)
        coef_str = input()
    coef = float(coef_str)
    return coef


def solve(a, b, c):
    D = b*b - 4*a*c
    roots = []
    if D > 0:
        t1 = (-b + D**0.5) / 2 / a
        roots += solve_square(t1)
        t2 = (-b - D**0.5) / 2 / a
        roots += solve_square(t2)
    elif D == 0:
        t = -b / 2 / a
        roots += solve_square(t)
    return roots


def solve_square(n):
    roots = []
    if n > 0:
        roots.append(n**0.5)
        roots.append(-n**0.5)
    elif n == 0:
        roots.append[n]
    return roots


a = get_coef(1, 'Введите коэффициент A:')
b = get_coef(2, 'Введите коэффициент B:')
c = get_coef(3, 'Введите коэффициент C:')

roots = solve(a, b, c)
roots.sort()
```

```
match len(roots):
    case 0:
        print("Не найдено действительных корней")
    case 1:
        print("Один корень: {}".format(roots[0]))
    case 2:
        print("Два корня: {}, {}".format(roots[0], roots[1]))
    case 3:
        print("Три корня: {}, {}, {}".format(roots[0], roots[1], roots[2],))
    case 4:
        print("Четыре корня: {}, {}, {}, {}".format(roots[0], roots[1],
roots[2], roots[3]))
```

## Вывод:

```
andrey@andrey-HP-EliteBook-840-G5:~/BKIT$ /bin/python3 /home/andrey/BKIT/lab_1/bisquared.py
Введите коэффициент A:
1
Введите коэффициент B:
-5
Введите коэффициент C:
4
Четыре корня: -2.0, -1.0, 1.0, 2.0
andrey@andrey-HP-EliteBook-840-G5:~/BKIT$ /bin/python3 /home/andrey/BKIT/lab_1/bisquared.py
Введите коэффициент A:
1
Введите коэффициент B:
1
Введите коэффициент C:
1
Не найдено действительных корней
andrey@andrey-HP-EliteBook-840-G5:~/BKIT$ /bin/python3 /home/andrey/BKIT/lab_1/bisquared.py 1 -2 1
Два корня: -1.0, 1.0
```

## Дополнительное задание 2
## Текст программы (Rust):

```rust
use std::env;
use std::io;


fn get_coef(index: usize, name: &str) -> f64{;
    match env::args().nth(index){
        Some(string) => return string.trim().parse().unwrap_or_else(|error|
{
            panic!("Incorrect argument value")
        }),
        None => println!("Enter {}:", name)
    }
    loop{
        let mut input: String = String::new();
        io::stdin().read_line(&mut input);
        match input.trim().parse(){
            Ok(value) => return value,
            Err(_) => println!("Try again")
        }
    }
}


fn solve(a: f64, b: f64, c: f64) -> [Option<f64>; 4]{
    let d = b*b - 4.0*a*c;
    let mut roots: [Option<f64>; 4] = [None, None, None, None];
    if d < 0.0 {return roots;}

    let x1 = (-b - d.sqrt())/ (2.0*a);
    let x2 = (-b + d.sqrt())/ (2.0*a);

    if x1 >= 0.0 {
        roots[0] = Some(x1.sqrt());
        if x1 > 0.0 {roots[1] = Some(-x1.sqrt());}
    }

    if x2 == x1 {return roots;}

    if x2 >= 0.0 {
        roots[2] = Some(x2.sqrt());
        if x2 > 0.0 {roots[3] = Some(-x2.sqrt());}
```

```
    }

    return roots;
}


fn main(){
    let a = get_coef(1, "a");
    let b = get_coef(2, "b");
    let c = get_coef(3, "c");

    let roots = solve(a, b, c);
    let mut count = 0;
    for i in 0..roots.len() {
        if !roots[i].is_none(){
            count += 1;
            println!("x{} = {}",count, roots[i].unwrap());
        }
    }
    if count == 0 {println!("No roots found");}
}
```

## Вывод:

```
andrey@andrey-HP-EliteBook-840-G5:~/BKIT/lab_1/rust$ ./main
Enter a:
1
Enter b:
-5
Enter c:
4
x1 = 1
x2 = -1
x3 = 2
x4 = -2
andrey@andrey-HP-EliteBook-840-G5:~/BKIT/lab_1/rust$ ./main 1
Enter b:
2
Enter c:
3
No roots found
andrey@andrey-HP-EliteBook-840-G5:~/BKIT/lab_1/rust$ ./main 1 -2 1
x1 = 1
x2 = -1
```