**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 №1
# «Основные конструкции языка Python.»

Выполнил:
студент группы ИУ5-31Б
Зобнин Александр
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Подпись и дата:

# Постановка задачи

## 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

## main.py

```
import sys
import math


def get_coef(index, prompt):
    '''
    Читаем коэффициент из командной строки или вводим с клавиатуры

    Args:
        index (int): Номер параметра в командной строке
        prompt (str): Приглашение для ввода коэффицента

    Returns:
        float: Коэффициент квадратного уравнения
    '''
    while True:
            try:
                # Пробуем прочитать коэффициент из командной строки
                coef_str = sys.argv[index]
            except:
                # Вводим с клавиатуры
                print(prompt)
                coef_str = input()
            # Переводим строку в действительное число
            try:
                coef = float(coef_str)
            except ValueError:
                print("Некорректный ввод. Попробуйте ещё раз.")
                continue
            else:
                break
    return coef


def get_roots(a, b, c):
    '''
    Вычисление корней квадратного уравнения

    Args:
        a (float): коэффициент A
        b (float): коэффициент B
        c (float): коэффициент C

    Returns:
        list[float]: Список корней
    '''
    result = []
    D = b * b - 4 * a * c
    if D == 0.0:
        root = -b / (2.0 * a)
        result.append(root)
    elif D > 0.0:
        sqD = math.sqrt(D)
        root1 = (-b + sqD) / (2.0 * a)
        root2 = (-b - sqD) / (2.0 * a)
        result.append(root1)
```

```
        result.append(root2)
    return result


def get_real_roots(result):
    real_result = []
    for root in result:
        if root > 0:
            real_result.append(math.sqrt(root))
            real_result.append(-math.sqrt(root))
        elif root == 0:
            real_result.append(abs(root))
    return real_result


def main():
    '''
    Основная функция
    '''
    a = get_coef(1, 'Введите коэффициент A:')
    b = get_coef(2, 'Введите коэффициент B:')
    c = get_coef(3, 'Введите коэффициент C:')
    # Вычисление корней
    roots = get_roots(a, b, c)
    real_roots = get_real_roots(roots)
    # Вывод корней
    len_roots = len(real_roots)
    if len_roots == 0:
        print('Нет корней')
    elif len_roots == 1:
        print('Один корень: {}'.format(real_roots[0]))
    elif len_roots == 2:
        print('Два корня: {} и {}'.format(real_roots[0], real_roots[1]))
    elif len_roots == 3:
        print('Три корня: {}, {} и {}'.format(real_roots[0], real_roots[1],
real_roots[2]))
    elif len_roots == 4:
        print('Четыре корня: {}, {}, {} и {}'.format(real_roots[0], real_roots[1],
real_roots[2], real_roots[3]))


# Если сценарий запущен из командной строки
if __name__ == "__main__":
    main()

# Примеры запуска
# roots_proc.py 1 0 10 (Нет корней)
# roots_proc.py 1 0 -4 (Два корня)
# roots_proc.py -4 16 0 (Три корня)
# roots_proc.py 1 -13 36 (Четыре корня)
```

## main.swift

```
import Foundation

// # Linear Equation Solver
func linearSolve(a: Double, b: Double) -> [Double] {
    if a == 0 {
        return []
    }

    return [Double(-b/a)]
}
```

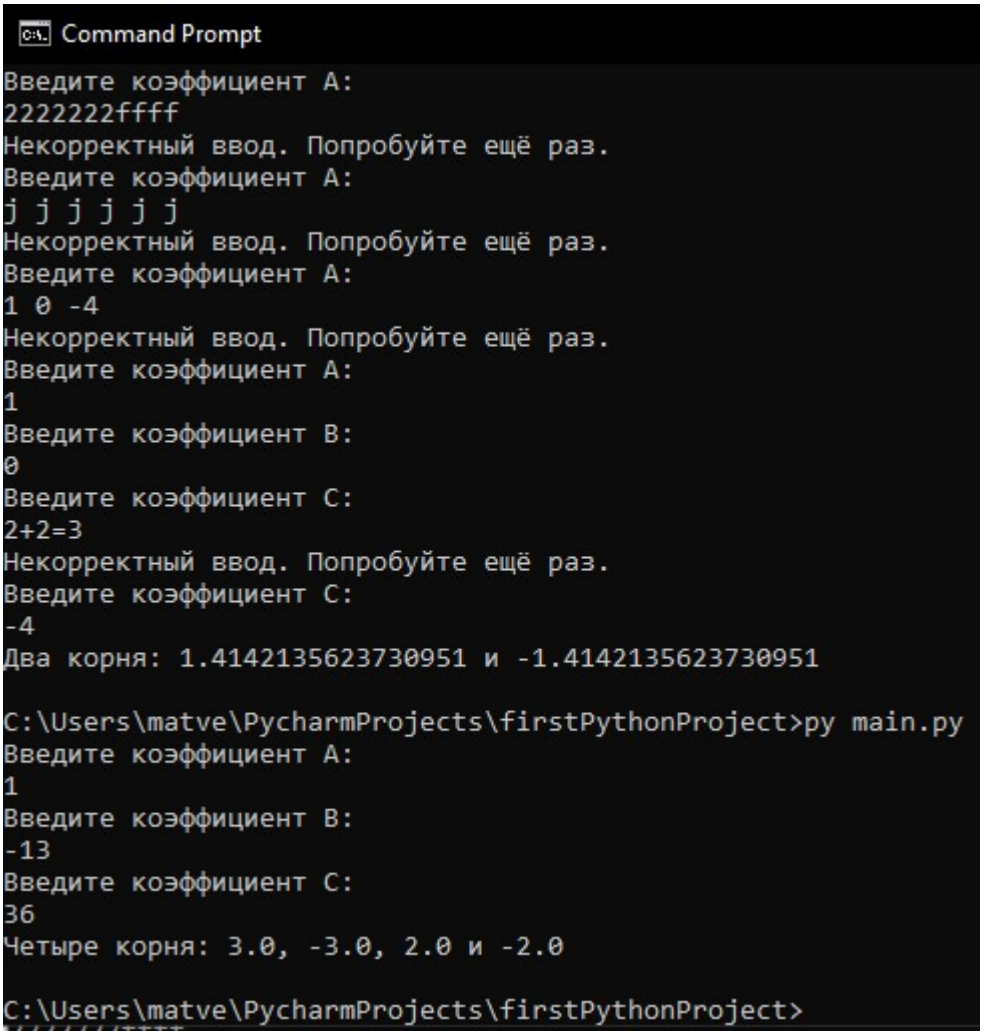
```
// # Quadratic Equation Solver
func quadraticSolve(a: Double, b: Double, c: Double, threshold: Double = 0.0001) -
> [Double] {
    if a == 0 { return linearSolve(a: b, b: c) }

    var roots = [Double]()

    var d = pow(b, 2) - 4*a*c

    // Check if discriminate is within the 0 threshold
    if -threshold < d && d < threshold { d = 0 }

    if d > 0 {
        let x_1 = Double((-b + sqrt(d))/(2*a))
        let x_2 = Double((-b - sqrt(d))/(2*a))
        roots = [x_1, x_2]
    } else if d == 0 {
        let x = Double(-b/(2*a))
        roots = [x, x]
    }

    return roots
}


// # Biquadratic Equation Solver
func biquadrateSolve(a: Double, b: Double, c: Double) -> [Double] {
    var result = [Double]()
    let solutions = quadraticSolve(a: a, b: b, c: c)
    for root in solutions {
        if root > 0 {
            result.append(-root.squareRoot())
            result.append(root.squareRoot())
        } else if root == 0 {
            result.append(0)
        }
    }
    return result
}


// Example of usage
let solutions = biquadrateSolve(a: 1, b: -13, c: 36)
print(solutions)
```

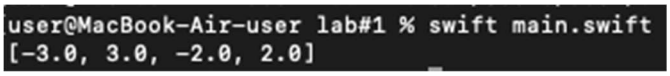
# Анализ результатов

```
Введите коэффициент A:
1
Введите коэффициент B:
0
Введите коэффициент C:
10
Нет корней

Process finished with exit code 0
```

```
Введите коэффициент A:
-4
Введите коэффициент B:
fsdg
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент B:
3415 1345 1345
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент B:
16
Введите коэффициент C:
0
Три корня: 0.0, 2.0 и -2.0

Process finished with exit code 0
```

```
Command Prompt
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\Users\matve>cd PycharmProjects

C:\Users\matve\PycharmProjects>cd firstPythonProject

C:\Users\matve\PycharmProjects\firstPythonProject>py main.py
Введите коэффициент A:
2222222ffff
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент A:
j j j j j j
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент A:
1 0 -4
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент A:
1
Введите коэффициент B:
0
Введите коэффициент C:
2+2=3
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент C:
-4
Два корня: 1.4142135623730951 и -1.4142135623730951

C:\Users\matve\PycharmProjects\firstPythonProject>
```


Command Prompt
Введите коэффициент A:
2222222ffff
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент A:
j j j j j j
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент A:
1 0 -4
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент A:
1
Введите коэффициент B:
0
Введите коэффициент C:
2+2=3
Некорректный ввод. Попробуйте ещё раз.
Введите коэффициент C:
-4
Два корня: 1.4142135623730951 и -1.4142135623730951
C:\Users\matve\PycharmProjects\firstPythonProject>py main.py
Введите коэффициент A:
1
Введите коэффициент B:
-13
Введите коэффициент C:
36
Четыре корня: 3.0, -3.0, 2.0 и -2.0
C:\Users\matve\PycharmProjects\firstPythonProject>


user@MacBook-Air-user lab#1 % swift main.swift
[-3.0, 3.0, -2.0, 2.0]