**Отчёт по лабораторной работе №1**

**Основные конструкции языка Python.**

**Цель лабораторной работы:** изучение основных конструкций языка Python.

Задание:

Разработать программу для решения биквадратного уравнения.

Программа должна быть разработана в виде консольного приложения на языке Python.

Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).

Коэффициенты А, В, С 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.

Если коэффициент А, В, С 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.

**Текст программы**

import sys

import math

class BiquadraticRoots:

    def \_\_init\_\_ (self):

        self.a = 0

        self.b = 0

        self.c = 0

        self.num\_roots = 0

        self.roots\_list = []

    def get\_coef(self, index, prompt):

        try:

            coef\_str = sys.argv[index]

        except:

            print(prompt)

            coef\_str = input()

        try:

            coef = float(coef\_str)

        except:

            print("Коэффициент должен быть числом")

            sys.exit(0)

    def get\_coefs(self):

        self.a = self.get\_coef(1, "Введите коэффициент a: ")

        self.b = self.get\_coef(2, "Введите коэффициент b: ")

        self.c = self.get\_coef(3, "Введите коэффициент c: ")

    def calculate\_roots(self):

        a = self.a

        b = self.b

        c = self.c

        D = b\*\*2 - 4\*a\*c

        if D == 0:

            if -b/(2\*a) > 0:

                root1 = math.sqrt(-b/(2\*a))

                root2 = -root1

                self.num\_roots = 2

                self.roots\_list.append(root1)

                self.roots\_list.append(root2)

            elif (a != 0) and (b == 0):

                root1 = 0

                self.roots\_list.append(root1)

        if D > 0:

            buf1 = (-b - math.sqrt(D))/(2\*a)

            buf2 = (-b + math.sqrt(D))/(2\*a)

            if buf1 > 0:

                root1 = math.sqrt(buf1)

                root2 = -root1

                self.num\_roots += 2

                self.roots\_list.append(root1)

                self.roots\_list.append(root2)

            elif buf1 == 0:

                root1 = 0

                self.num\_roots += 1

                self.roots\_list.append(root1)

            if buf2 > 0:

                root3 = math.sqrt(buf2)

                root4 = -root3

                self.num\_roots += 2

                self.roots\_list.append(root3)

                self.roots\_list.append(root4)

            elif buf2 == 0:

                root3 = 0

                self.num\_roots += 1

                self.roots\_list.append(root3)

    def print\_roots(self):

        if self.num\_roots != len(self.roots\_list):

            print(('Ошибка. Уравнение содержит {} действительных корней, ' +\

                'но было вычислено {} корней.').format(self.num\_roots, len(self.roots\_list)))

        else:

            if self.num\_roots == 0:

                print('Корней нет')

            elif self.num\_roots == 1:

                print('Один корень: {}'.format(self.roots\_list[0]))

            elif self.num\_roots == 2:

                print('Два корня: {} и {}'.format(self.roots\_list[0], \

                    self.roots\_list[1]))

            elif self.num\_roots == 3:

                print('Три корня: {} и {} и {}'.format(self.roots\_list[0], self.roots\_list[1], self.roots\_list[2]))

            else:

                print('Четыре корня: {} и {} и {} и {}'.format(self.roots\_list[0], self.roots\_list[1], \

                self.roots\_list[2], self.roots\_list[3]))

def main():

    r = BiquadraticRoots()

    r.get\_coefs()

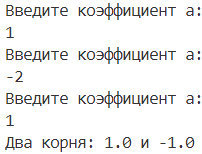
    r.calculate\_roots()

    r.print\_roots()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Пример выполнения программы**

****