**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №1

«Основные конструкции языка Python»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32Б: |  | преподаватель каф. ИУ5 |
| Бекетов Роман Александрович |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2022 г.

**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.

**Листинг**

import sys

import math

class BisquareRoots:

def \_\_init\_\_(self):

'''

Конструктор класса

'''

# Объявление коэффициентов

self.coef\_A = 0.0

self.coef\_B = 0.0

self.coef\_C = 0.0

# Количество корней

self.num\_roots = 0

# Список корней

self.roots\_list = []

def get\_coef(self, 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:

# Пробуем прочитать коэффициент из командной строки

try:

coef = float(sys.argv[index])

if coef == 0.0 and index == 1:

raise UserWarning

break

except Exception as e:

# Вводим с клавиатуры

try:

if (e.\_\_class\_\_ == ValueError):

print('Некорректный ввод.')

if (e.\_\_class\_\_ == UserWarning):

print('Введённое значение параметра А некорректно');

print(prompt)

coef = float(input())

if (index == 1 and coef == 0.0):

raise UserWarning

break

except ValueError:

print('Некорректный ввод.')

except UserWarning:

print('Введённое значение параметра А некорректно');

except:

print('Что-то пошло не так...')

return coef

def get\_coefs(self):

'''

Чтение трех коэффициентов

'''

self.coef\_A = self.get\_coef(1, 'Введите коэффициент А:')

self.coef\_B = self.get\_coef(2, 'Введите коэффициент B:')

self.coef\_C = self.get\_coef(3, 'Введите коэффициент C:')

def calculate\_roots(self):

'''

Вычисление корней квадратного уравнения

'''

a = self.coef\_A

b = self.coef\_B

c = self.coef\_C

# Вычисление дискриминанта и корней

D = b\*b - 4\*a\*c

if D == 0.0:

root = -b / (2.0\*a)

if root > 0.0:

self.num\_roots = 2

self.roots\_list.append(math.sqrt(root))

self.roots\_list.append(-math.sqrt(root))

elif root == 0:

self.num\_roots = 1

self.roots\_list.append(root)

elif D > 0.0:

sqD = math.sqrt(D)

root1 = (-b + sqD) / (2.0\*a)

root2 = (-b - sqD) / (2.0\*a)

if root1 > 0.0:

self.num\_roots += 2

self.roots\_list.append(math.sqrt(root1))

self.roots\_list.append(-math.sqrt(root1))

elif root1 == 0.0:

self.num\_roots += 1

self.roots\_list.append(root1)

if root2 > 0.0:

self.num\_roots += 2

self.roots\_list.append(math.sqrt(root2))

self.roots\_list.append(-math.sqrt(root2))

elif root2 == 0.0:

self.num\_roots += 1

self.roots\_list.append(root2)

def print\_roots(self):

# Проверка отсутствия ошибок при вычислении корней

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:

match self.num\_roots:

case 0:

print('Нет корней')

case 1:

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

case 2:

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

case 3:

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

case 4:

print('Четыре корня: {}, {}, {} и {}'.format(self.roots\_list[0], self.roots\_list[1], self.roots\_list[2], self.roots\_list[3]))

case \_:

print('Ошибка выполнения.')

def main():

'''

Основная функция

'''

# Создание объекта класса

r = BisquareRoots()

# Последовательный вызов необходимых методов

r.get\_coefs()

r.calculate\_roots()

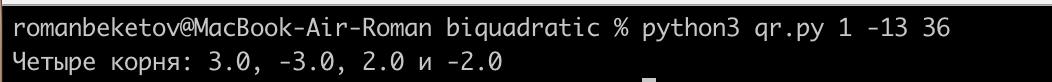
r.print\_roots()

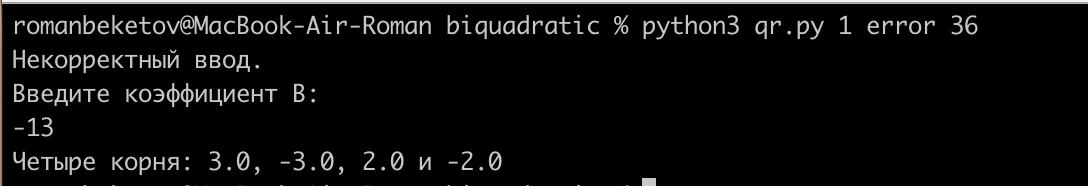
# Если сценарий запущен из командной строки

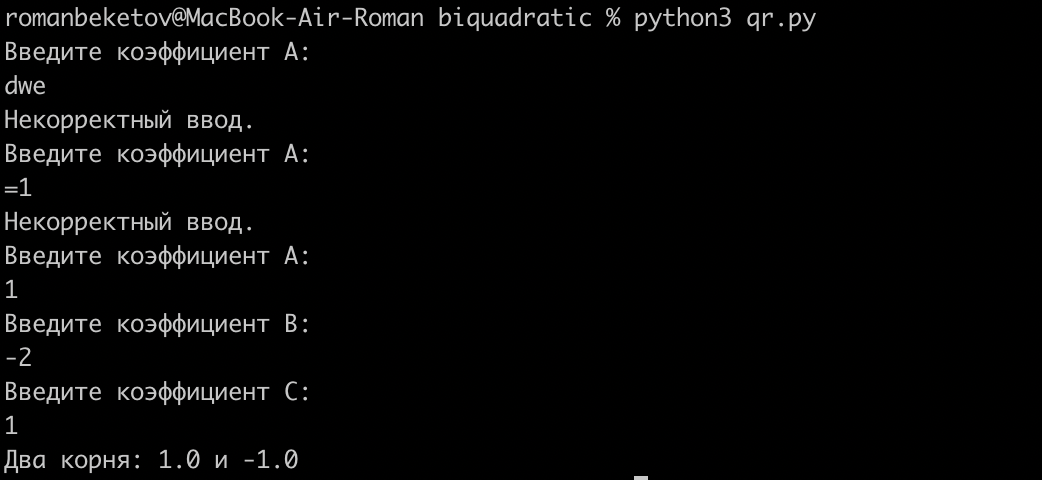
if \_\_name\_\_ == "\_\_main\_\_":

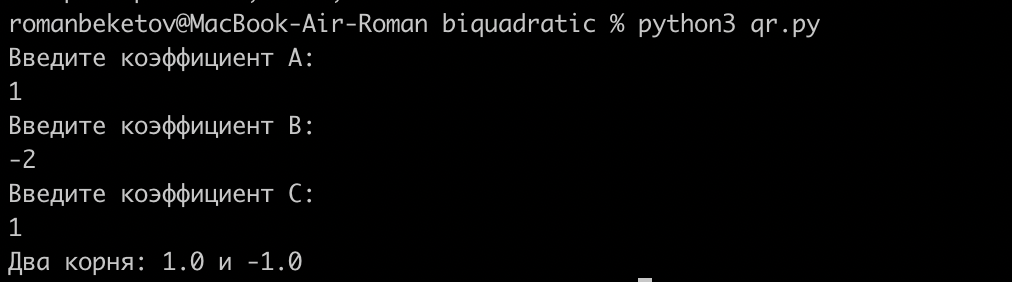
main()

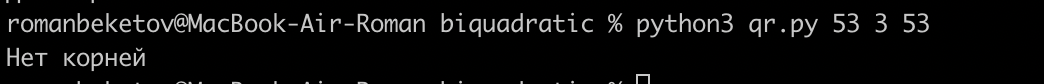
**Тесты**

****

****

****

****

****