**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет информатика и системы управления

Кафедра системы обработки информации и управления

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №1

«Основные конструкции языка Python»

Вариант 21

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32Б: |  | преподаватель каф. ИУ5 |
| Угрюмов Н.А. |  | Нардид А.Н. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2025 г.

**Задание**

Разработать программу для решения [биквадратного уравнения.](https://ru.wikipedia.org/wiki/%D0%A3%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5_%D1%87%D0%B5%D1%82%D0%B2%D1%91%D1%80%D1%82%D0%BE%D0%B9_%D1%81%D1%82%D0%B5%D0%BF%D0%B5%D0%BD%D0%B8#%D0%91%D0%B8%D0%BA%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D0%BD%D0%BE%D0%B5_%D1%83%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5)

1. Программа должна быть разработана в виде консольного приложения на языке Python.
2. Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).
3. Коэффициенты А, В, С могут быть заданы в виде параметров командной строки ( [вариант задания параметров приведен в конце файла с примером кода](https://github.com/ugapanyuk/BKIT_2022/blob/main/code/lab1_code) ). Если они не заданы, то вводятся с клавиатуры в соответствии с пунктом 2. [Описание работы с параметрами командной строки.](https://realpython.com/python-command-line-arguments/#the-command-line-interface)
4. 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.
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 С++).

**Текст программы**

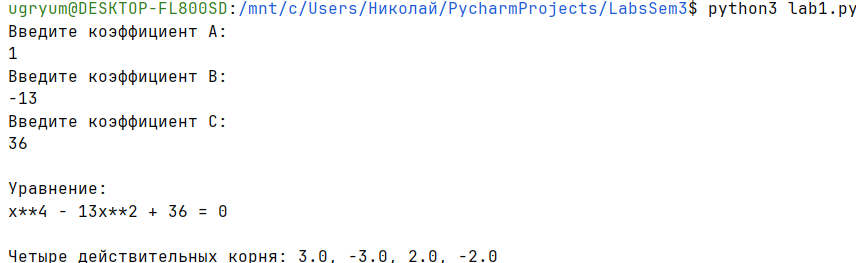
* **Файл lab1.py** (решение с применением процедурной парадигмы):

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  
 '''  
  
 # Сначала пробуем получить из командной строки* if index < len(sys.argv):  
 try:  
 coef = float(sys.argv[index])  
 if index == 1 and coef == 0.0:  
 print("Коэффициент А не может быть равен 0 для биквадратного уравнения. Введите снова.")  
 else:  
 print(f"{prompt.split(':')[0]}: {sys.argv[index]}")  
 return coef  
 except (ValueError, IndexError):  
 print("Некорректное значение коэффициента в командной строке.")  
  
 *# Если не получилось из командной строки, вводим с клавиатуры* while True:  
 try:  
 print(prompt)  
 coef\_str = input()  
 coef = float(coef\_str)  
 if index == 1 and coef == 0.0:  
 print("Коэффициент А не может быть равен 0 для биквадратного уравнения. Введите снова.")  
 continue  
 return coef  
 except ValueError:  
 print("Некорректный ввод, попробуйте снова.")  
  
  
def get\_roots(a, b, c):  
 *'''  
 Вычисление корней квадратного уравнения  
  
 Args:  
 a (float): коэффициент А  
 b (float): коэффициент B  
 c (float): коэффициент C  
  
 Returns:  
 list[float]: Список корней  
 '''  
  
 # Списки для итоговых корней и корней t = x\*\*2, соответственно* result = []  
 t\_roots = []  
  
 *# Решаем отностительно t = x\*\*2* D = b \* b - 4 \* a \* c  
  
 if D < 0:  
 *# Нет корней биквадратного уравнения* return result  
  
 if D == 0.0:  
 t = -b / (2.0 \* a)  
 t\_roots.append(t)  
 elif D > 0.0:  
 sqD = math.sqrt(D)  
 t1 = (-b + sqD) / (2.0 \* a)  
 t2 = (-b - sqD) / (2.0 \* a)  
 t\_roots.append(t1)  
 t\_roots.append(t2)  
 *# Случай с отрицательным D нет смысла обрабатывать, нет корней  
  
 # Находим корни x из t = x\*\*2* for t in t\_roots:  
 if t > 0:  
 x\_abs = math.sqrt(t)  
 result.extend([x\_abs, -x\_abs])  
 elif t == 0:  
 result.append(0.0)  
  
 result = [round(i, 2) for i in result]  
 return result  
  
def print\_equation(a, b, c):  
 *# Вывод уравнения на экран* a = int(a) if a == int(a) else a  
 b = int(b) if b == int(b) else b  
 c = int(c) if c == int(c) else c  
  
 print("\nУравнение: ")  
  
 summand1 = ("" if a >= 0 else "-") + (str(abs(a)) if abs(a) != 1 else "") + "x\*\*4"  
 summand2 = ((" + " if b > 0 else " - ") + (str(abs(b)) if abs(b) != 1 else "") + "x\*\*2") if b else ""  
 summand3 = ((" + " if c > 0 else " - ") + str(abs(c))) if c else ""  
 equation = summand1 + summand2 + summand3 + " = 0"  
  
 print(equation + "\n")  
  
def main():  
 *'''  
 Основная функция  
 '''* while True:  
 a = get\_coef(1, "Введите коэффициент А:")  
 if a != 0:  
 break  
 print("Коэффициент А не может быть равен 0 для биквадратного уравнения. Введите снова.")  
  
 b = get\_coef(2, "Введите коэффициент B:")  
 c = get\_coef(3, "Введите коэффициент C:")  
  
 *# Вычисление корней* roots = get\_roots(a, b, c)  
  
 *# Вывод уравнения* print\_equation(a, b, c)  
  
 *# Вывод корней* len\_roots = len(roots)  
 if len\_roots == 0:  
 print("Нет действительных корней")  
 elif len\_roots == 1:  
 print("Один действительный корень: {}".format(roots[0]))  
 elif len\_roots == 2:  
 print("Два действительных корня: {} и {}".format(roots[0], roots[1]))  
 elif len\_roots == 3:  
 print("Три действительных корня: {}, {}, {}".format(roots[0], roots[1], roots[2]))  
 elif len\_roots == 4:  
 print("Четыре действительных корня: {}, {}, {}, {}".format(roots[0], roots[1], roots[2], roots[3]))  
  
  
*# Если сценарий запущен из командной строки*if \_\_name\_\_ == "\_\_main\_\_":  
 main()

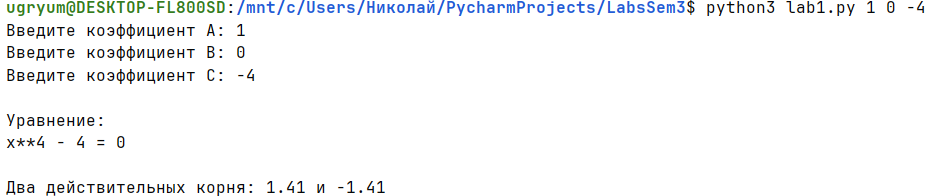
* **Файл lab1\_OOP\_solution.py** (решение с применением объектно-ориентированной парадигмы):

import sys  
import math  
  
class BiquadraticSolver:  
 def \_\_init\_\_(self, a, b, c):  
 self.a = a  
 self.b = b  
 self.c = c  
  
 def print\_equation(self):  
 *# Вывод уравнения на экран* a = int(self.a) if self.a == int(self.a) else self.a  
 b = int(self.b) if self.b == int(self.b) else self.b  
 c = int(self.c) if self.c == int(self.c) else self.c  
  
 print("\nУравнение: ")  
  
 summand1 = ("" if a >= 0 else "-") + (str(abs(a)) if abs(a) != 1 else "") + "x\*\*4"  
 summand2 = ((" + " if b > 0 else " - ") + (str(abs(b)) if abs(b) != 1 else "") + "x\*\*2") if b else ""  
 summand3 = ((" + " if c > 0 else " - ") + str(abs(c))) if c else ""  
 equation = summand1 + summand2 + summand3 + " = 0"  
  
 print(equation + "\n")  
  
  
 def solve(self):  
 a, b, c = self.a, self.b, self.c  
  
 *# Списки для итоговых корней и корней t = x\*\*2, соответственно* result = []  
 t\_roots = []  
  
 *# Решаем отностительно t = x\*\*2* D = b \* b - 4 \* a \* c  
  
 if D < 0:  
 *# Нет корней биквадратного уравнения* return result  
  
 if D == 0.0:  
 t = -b / (2.0 \* a)  
 t\_roots.append(t)  
 elif D > 0.0:  
 sqD = math.sqrt(D)  
 t1 = (-b + sqD) / (2.0 \* a)  
 t2 = (-b - sqD) / (2.0 \* a)  
 t\_roots.append(t1)  
 t\_roots.append(t2)  
 *# Случай с отрицательным D нет смысла обрабатывать, нет корней  
  
 # Находим корни x из t = x\*\*2* for t in t\_roots:  
 if t > 0:  
 x\_abs = math.sqrt(t)  
 result.extend([x\_abs, -x\_abs])  
 elif t == 0:  
 result.append(0.0)  
  
 result = [round(i, 2) for i in result]  
 return result  
  
 def print\_solution(self):  
 roots = self.solve()  
  
 *# Вывод корней* len\_roots = len(roots)  
 if len\_roots == 0:  
 print("Нет действительных корней")  
 elif len\_roots == 1:  
 print("Один действительный корень: {}".format(roots[0]))  
 elif len\_roots == 2:  
 print("Два действительных корня: {} и {}".format(roots[0], roots[1]))  
 elif len\_roots == 3:  
 print("Три действительных корня: {}, {}, {}".format(roots[0], roots[1], roots[2]))  
 elif len\_roots == 4:  
 print("Четыре действительных корня: {}, {}, {}, {}".format(roots[0], roots[1], roots[2], roots[3]))  
  
  
class InputHandler:  
 @staticmethod  
 def get\_coef(index, prompt):  
 if index < len(sys.argv):  
 try:  
 coef = float(sys.argv[index])  
 if index == 1 and coef == 0.0:  
 print("Коэффициент А не может быть равен 0 для биквадратного уравнения. Введите снова.")  
 else:  
 print(f"{prompt.split(':')[0]}: {sys.argv[index]}")  
 return coef  
 except (ValueError, IndexError):  
 print("Некорректное значение коэффициента в командной строке.")  
  
 *# Если не получилось из командной строки, вводим с клавиатуры* while True:  
 try:  
 print(prompt)  
 coef\_str = input()  
 coef = float(coef\_str)  
 if index == 1 and coef == 0.0:  
 print("Коэффициент А не может быть равен 0 для биквадратного уравнения. Введите снова.")  
 continue  
 return coef  
 except ValueError:  
 print("Некорректный ввод, попробуйте снова.")  
  
def main():  
 a = InputHandler.get\_coef(1, "Введите коэффициент А:")  
 b = InputHandler.get\_coef(2, "Введите коэффициент B:")  
 c = InputHandler.get\_coef(3, "Введите коэффициент C:")  
  
 solver = BiquadraticSolver(a, b, c)  
 solver.print\_equation()  
 solver.print\_solution()  
  
*# Если сценарий запущен из командной строки*if \_\_name\_\_ == "\_\_main\_\_":  
 main()

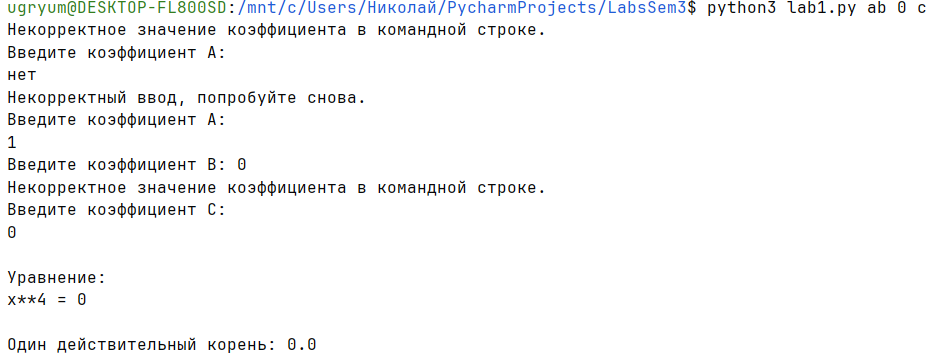
**Скриншоты работы приложения**



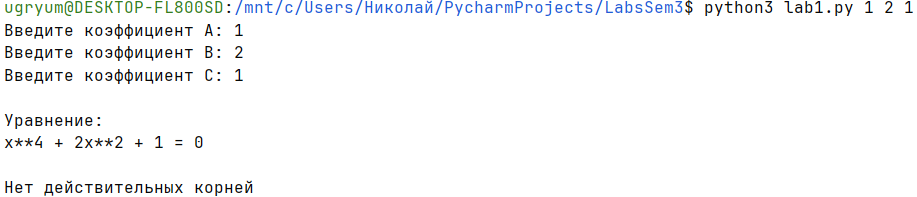
**Рис. 1:** решение уравнения с 4-мя действительными корнями и вводом коэффициентов через консоль



**Рис. 2:** решение уравнения с 2-мя действительными корнями и указанием коэффициентов через аргументы командной строки



**Рис. 3:** обработка некорректного ввода



**Рис. 4:** решение уравнения, не имеющего действительных корней

**Ветка с ЛР в моём GitHub:** <https://github.com/ugryum1/LabsSem3/tree/lab1>