Московский государственный технический

университет им. Н.Э. Баумана

Проверил:

Преподаватель каф. ИУ5

Подпись и дата:

Факультет «Информатика и система управления»

Кафедра ИУ5 «Система обработки информации и управления»

Выполнил:

Студент группы ИУ5-36Б

Яншин Т. И.

Подпись и дата:

Курс «Парадигмы и конструкции языков программирования»

Отчёт по лабораторной работе №1

**Основные конструкции языка Python.**

**Ссылка на репозиторий: https://github.com/reduct0r/Python\_lab1**

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

**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++).

**Код программы:**  
*Процедурная парадигма*  
main.py

import sys

import math

def get\_coefficient\_from\_input(prompt):

while True:

try:

value = float(input(prompt))

return value

except ValueError:

print("Некорректное значение. Попробуйте снова.")

def get\_coefficient(arg, prompt):

try:

return float(arg)

except (ValueError, TypeError):

print(f"Некорректное значение для {prompt} в командной строке.")

return get\_coefficient\_from\_input(prompt)

def get\_coefficients():

if len(sys.argv) == 4:

A = get\_coefficient(sys.argv[1], "коэффициент A")

B = get\_coefficient(sys.argv[2], "коэффициент B")

C = get\_coefficient(sys.argv[3], "коэффициент C")

else:

while True:

A = get\_coefficient\_from\_input("Введите коэффициент A: ")

if A != 0:

break

print("Коэффициент A не должен быть равен нулю. Попробуйте снова.")

B = get\_coefficient\_from\_input("Введите коэффициент B: ")

C = get\_coefficient\_from\_input("Введите коэффициент C: ")

return A, B, C

def solve\_quadratic(A, B, C):

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 solve\_biquadratic(A, B, C):

quadratic\_roots = solve\_quadratic(A, B, C)

biquadratic\_roots = set() # Используем set для исключения дубликатов

for root in quadratic\_roots:

if root >= 0:

biquadratic\_roots.add(math.sqrt(root))

biquadratic\_roots.add(-math.sqrt(root))

return list(biquadratic\_roots) # Преобразуем set обратно в список для печати

def main():

A, B, C = get\_coefficients()

print(f"Коэффициенты: A={A}, B={B}, C={C}")

print(f"Уравнение: {A}\*x^4 + {B}\*x^2 + {C} = 0")

result = solve\_biquadratic(A, B, C)

if len(result) == 0:

print("Нет корней")

else:

result = sorted(result) # Сортируем результат для наглядного вывода

print(f"Решение: {', '.join(map(str, result))}")

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

main()

*С использованием ООП:*

*oop.py*

import sys

import math

class BiquadSolver:

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

self.A = None

self.B = None

self.C = None

def get\_coefficient\_from\_input(self, prompt):

while True:

try:

value = float(input(prompt))

return value

except ValueError:

print("Некорректное значение. Попробуйте снова.")

def get\_coefficient(self, arg, prompt):

try:

return float(arg)

except (ValueError, TypeError):

print(f"Некорректное значение для {prompt} в командной строке.")

return self.get\_coefficient\_from\_input(prompt)

def get\_coefficients(self):

if len(sys.argv) == 4:

self.A = self.get\_coefficient(sys.argv[1], "коэффициент A")

self.B = self.get\_coefficient(sys.argv[2], "коэффициент B")

self.C = self.get\_coefficient(sys.argv[3], "коэффициент C")

else:

while True:

self.A = self.get\_coefficient\_from\_input("Введите коэффициент A: ")

if self.A != 0:

break

print("Коэффициент A не должен быть равен нулю. Попробуйте снова.")

self.B = self.get\_coefficient\_from\_input("Введите коэффициент B: ")

self.C = self.get\_coefficient\_from\_input("Введите коэффициент C: ")

def solve\_quadratic(self, A, B, C):

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 solve\_biquadratic(self):

quadratic\_roots = self.solve\_quadratic(self.A, self.B, self.C)

biquadratic\_roots = set() # Используем set для исключения дубликатов

for root in quadratic\_roots:

if root > 0:

biquadratic\_roots.add(math.sqrt(root))

biquadratic\_roots.add(-math.sqrt(root))

elif root == 0:

biquadratic\_roots.add(math.sqrt(root))

return list(biquadratic\_roots) # Преобразуем set обратно в список

def main(self):

self.get\_coefficients()

print(f"Коэффициенты: A={self.A}, B={self.B}, C={self.C}")

print(f"Уравнение: {self.A}\*x^4 + {self.B}\*x^2 + {self.C} = 0")

result = self.solve\_biquadratic()

if len(result) == 0:

print("Нет корней")

else:

print(f"Решение: {', '.join(map(str, result))}")

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

solver = BiquadSolver()

solver.main()

Экранные формы с примерами выполнения программы

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |