|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет имени Н.Э. Баумана**  **(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)** |

**Факультет «Информатика и системы управления»**

**Кафедра «Системы обработки информации и управления»**

Новицкий Ярослав ИУ5-35Б

Парадигмы и конструкции языков программирования

**ОТЧЁТ ПО**

**Лабораторной работе №1**

Москва

2023

**Задание.**

Задача: написать консольное приложение на Python для решения биквадратного уравнения. Программа должна выполнять следующие шаги:

* Получить коэффициенты A, B и C от пользователя через ввод с клавиатуры или параметры командной строки.
* Проверить, что введенные коэффициенты могут быть корректно преобразованы в действительные числа. Если какой-то коэффициент введен некорректно, программа должна запросить ввод повторно, пока значения не будут корректными.
* Вычислить дискриминант и определить, какие корни уравнения существуют в зависимости от значения дискриминанта.
* Если дискриминант положителен, вывести два действительных корня, еслили дискриминант равен нулю, вывести один действительный корень (корень кратности 2), если дискриминант отрицателен, вывести сообщение о том, что уравнение не имеет действительных корней.

Дополнительное задание: реализовать две версии программы на Python - одну с использованием процедурной парадигмы и другую с использованием объектно-ориентированной парадигмы.

**Текст программы.**

Процедурный подход:

import math  
import sys  
def get\_coef(index, line):  
 try:

i\_coef = sys.argv[index]

coef = float(i\_coef)

return coef

except:

print(line)

while 1:

try:

i\_coef = float(input())

break

except(ValueError, IndexError):

print("Введите корректное число для коэффициента.")

return i\_coef  
  
def roots(A, B, C):  
 if A == 0:  
 if B == 0:  
 if C == 0:  
 print("Любое число")  
 return  
 else:  
 print("Нет корней")  
 return  
 else:  
 xx = (-1) \* C / B  
 if xx < 0:  
 print("Нет корней")  
 return  
 else:  
 roots = [math.sqrt(xx), (-1) \* math.sqrt(xx)]  
 else:  
 D = B \*\* 2 - 4 \* A \* C  
 if D < 0:  
 print("Нет корней")  
 return  
 else:  
 x1 = ((-1) \* B - math.sqrt(D)) / (2 \* A)  
 x2 = ((-1) \* B + math.sqrt(D)) / (2 \* A)  
 roots = []  
 if x1 >= 0:  
 roots.append(math.sqrt(x1))  
 roots.append(-math.sqrt(x1))  
 if x2 >= 0:  
 roots.append(math.sqrt(x2))  
 roots.append(-math.sqrt(x2))  
 if not roots:  
 print("Нет корней")  
 return  
 unique\_roots = list(set(roots))  
 for root in unique\_roots:  
 print(root)  
  
def main():  
 A = get\_coef(1, 'Введите коэффициент A: ')  
 B = get\_coef(2, 'Введите коэффициент B: ')  
 C = get\_coef(3, 'Введите коэффициент C: ')  
 roots(A, B, C)

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

main()

Объектно-ориентированный подход:

import math  
import sys  
class roots:  
 def init(self, A = 0.0, B = 0.0, C = 0.0):  
 self.coef\_A = A  
 self.coef\_B = B  
 self.coef\_C = C  
 self.n\_roots = 0  
 self.roots = []  
 def get\_coef(self, index, line):

try:

i\_coef = sys.argv[index]

coef = float(i\_coef)

return coef

except:

print(line)

while 1:

try:

i\_coef = float(input())

break

except(ValueError, IndexError):

print("Введите корректное число для коэффициента.")

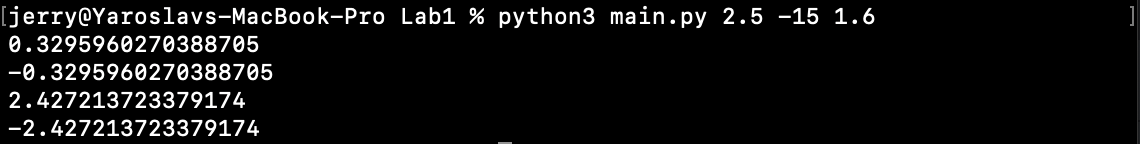
return i\_coef  
 def get\_coefs(self):  
 self.coef\_A = self.get\_coef(1, 'Введите коэффициент А:')  
 self.coef\_B = self.get\_coef(2, 'Введите коэффициент B:')  
 self.coef\_C = self.get\_coef(3, 'Введите коэффициент C:')  
 def c\_roots(self):  
 if self.coef\_A == 0:  
 if self.coef\_B == 0:  
 if self.coef\_C == 0:  
 print("Любое число")  
 return  
 else:  
 print("Нет корней")  
 return  
 else:  
 xx = (-1) \* self.coef\_C / self.coef\_B  
 if xx < 0:  
 print("Нет корней")  
 return  
 else:  
 roots = [math.sqrt(xx), (-1) \* math.sqrt(xx)]  
 else:  
 D = self.coef\_B \*\* 2 - 4 \* self.coef\_A \* self.coef\_C  
 if D < 0:  
 print("Нет корней")  
 return  
 else:  
 x1 = ((-1) \* self.coef\_B - math.sqrt(D)) / (2 \* self.coef\_A)  
 x2 = ((-1) \* self.coef\_B + math.sqrt(D)) / (2 \* self.coef\_A)  
 if x1 >= 0:  
 self.roots.append(math.sqrt(x1))  
 self.roots.append(-math.sqrt(x1))  
 if x2 >= 0:  
 self.roots.append(math.sqrt(x2))  
 self.roots.append(-math.sqrt(x2))  
 if not self.roots:  
 print("Нет корней")  
 return  
 unique\_roots = list(set(self.roots))  
 for root in unique\_roots:  
 print(root)  
def main():  
 r = roots()  
 r.get\_coefs()  
 r.c\_roots()  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**Экранные формы:**

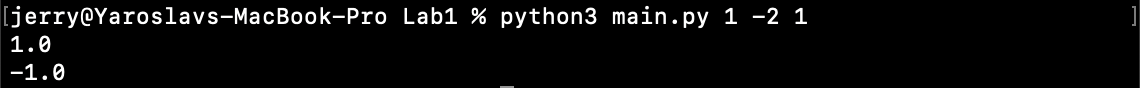
Любое число:



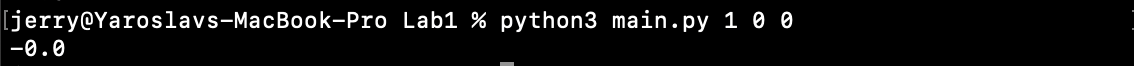
Четыре корня:



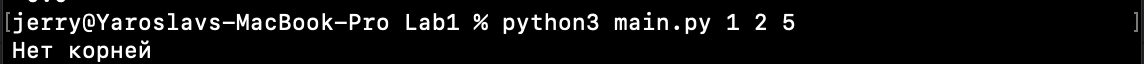
Два корня:



Один корень:



Нет корней:



Некорректный ввод из командной строки:

