МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

Факультет компьютерных технологий и прикладной математики

**Кафедра информационных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №1**

**МЕТОДЫ НАХОЖДЕНИЯ МИНИМУМА ФУНКЦИИ**

**ОДНОЙ ПЕРЕМЕННОЙ**

Работу выполнил А.А. Козин

(подпись)

Направление подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем

Направленность Программирование и информационные технологии

Руководитель

преп.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.С. Черная

(подпись)

Краснодар

2023

Оглавление

[Введение 3](#_Toc129606266)

[Индивидуальное задание 1 4](#_Toc129606267)

[Индивидуальное задание 2 6](#_Toc129606268)

[Индивидуальное задание 3 8](#_Toc129606269)

[Вывод 10](#_Toc129606270)

[ПРИЛОЖЕНИЕ 11](#_Toc129606271)

# **Введение**

Постановка задачи:

Для нахождения минимума/максимума функции от одной переменной была задана функция:

С отрезком , где

**Индивидуальное задание 1** – метод дихотомии.

**Индивидуальное задание 2** – метод золотого сечения.

**Индивидуальное задание 3** – метод Фибоначчи.

Ручное решение будет проделано до 3 шага в методе дихотомии, а в методах золотого сечения и Фибоначчи до 3 шага включительно.

Программное решение выполнено на языке программирования Python, в этом разделе будет приложен вывод терминала.

В конце отчета будет приложение с кодом.

# **Индивидуальное задание 1**

*Задание*: найти минимум функции одной переменной методом Дихотомии.

Ручное решение:

Изображение выглядит как текст

Автоматически созданное описание

Программное решение:

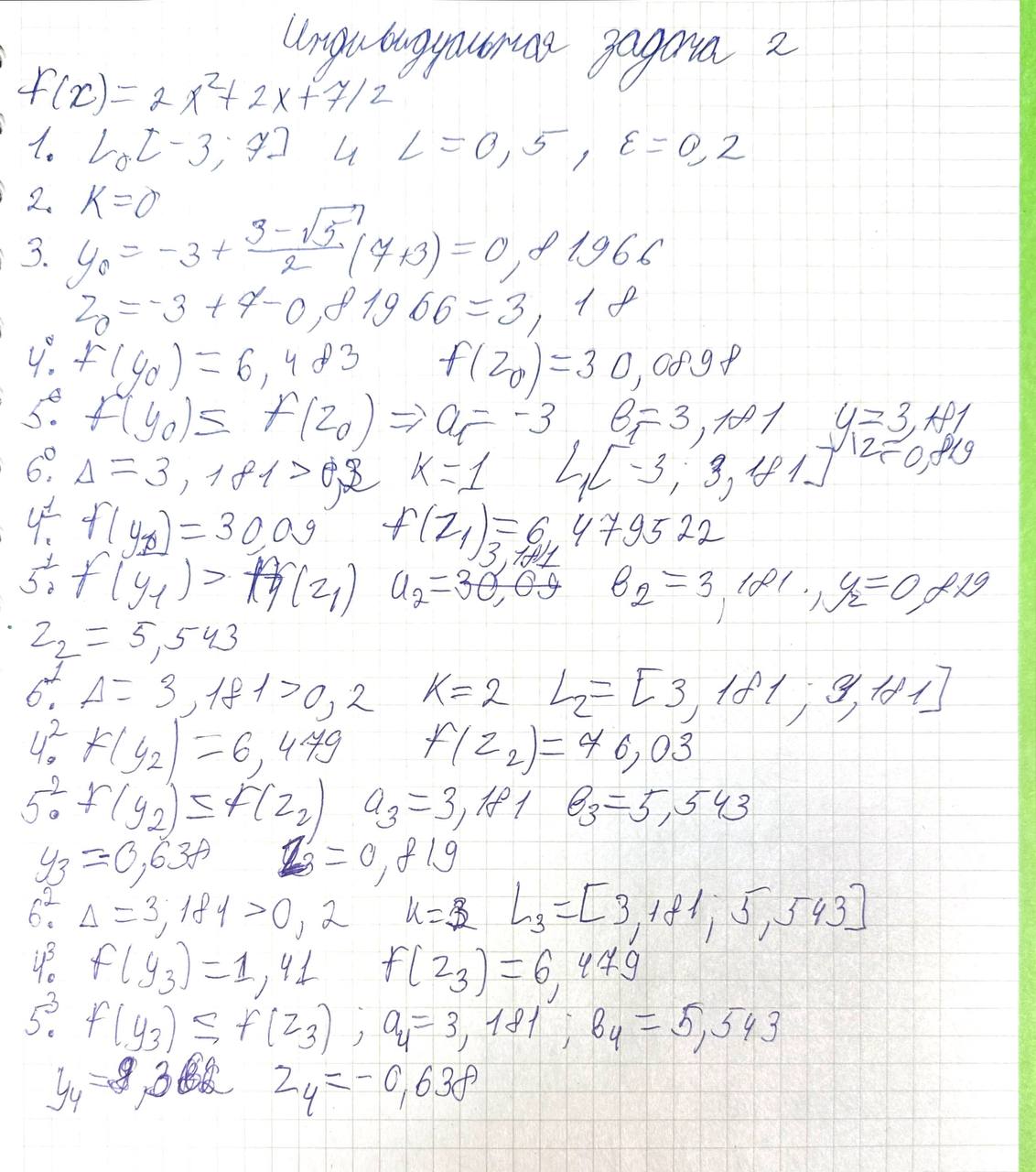
Изображение выглядит как текст

Автоматически созданное описание

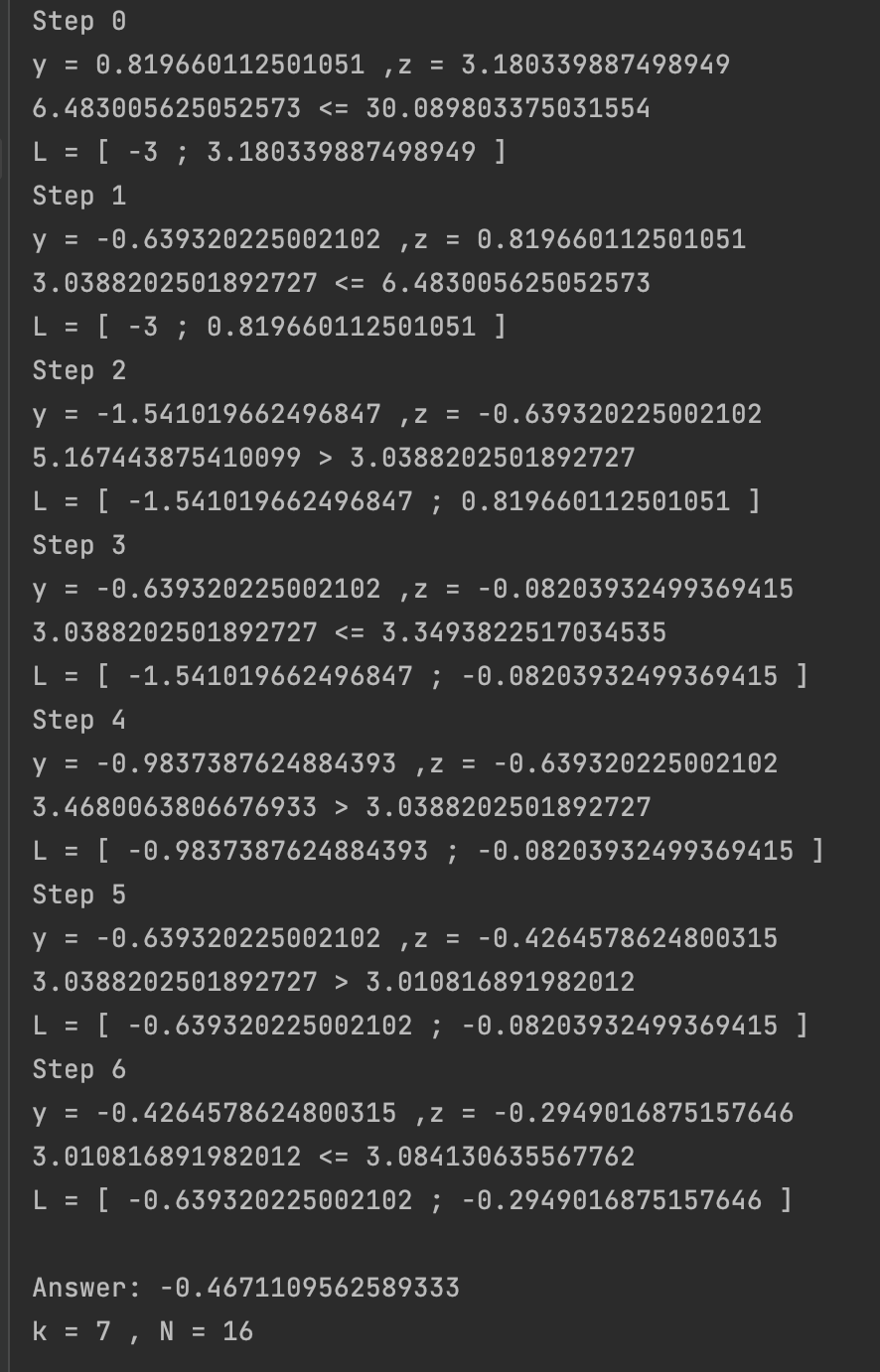
# **Индивидуальное задание 2**

*Задание*: найти минимум функции одной переменной методом золотого сечения.

Ручное решение:



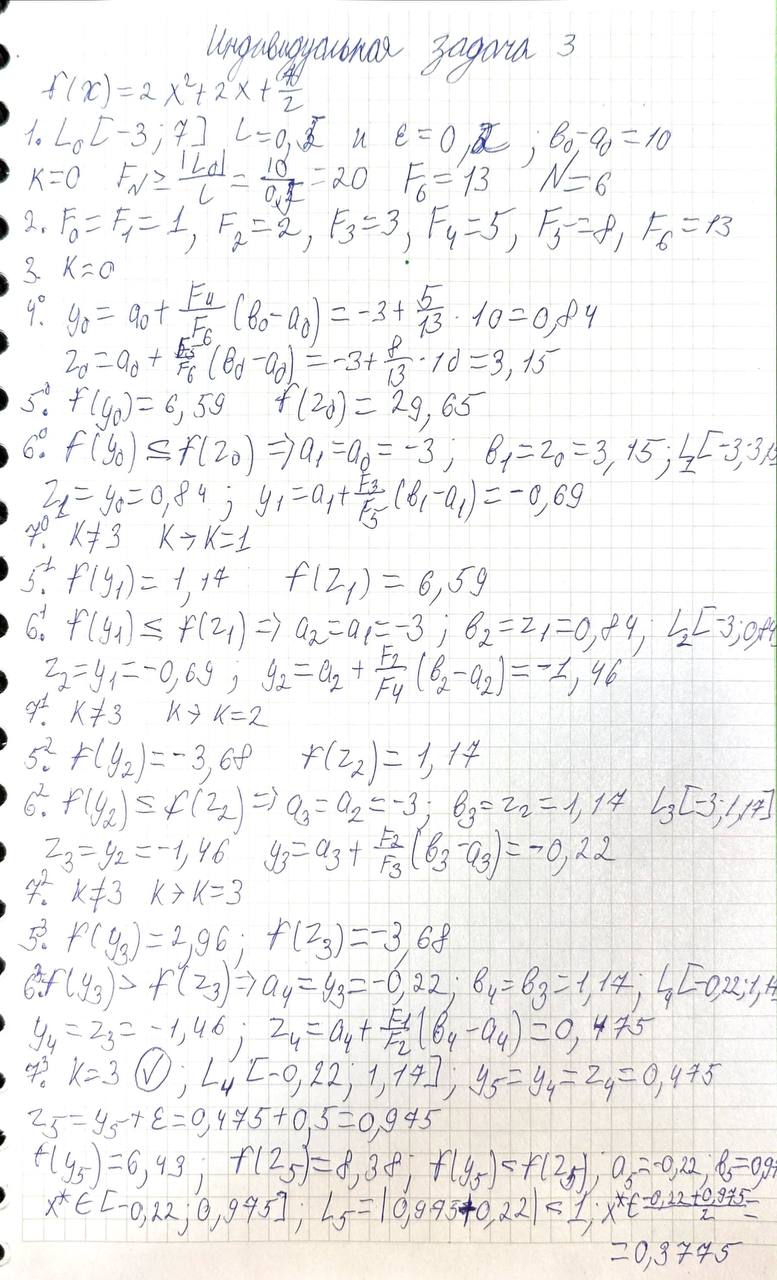
Программное решение:



# **Индивидуальное задание 3**

*Задание*: найти минимум функции одной переменной методом Фибоначчи.

Ручное решение:



Программное решение:

Изображение выглядит как текст

Автоматически созданное описание

# **Вывод**

У каждого из методов рассмотренных выше есть свои плюсы и минусы.

У метода Дихотомии:

* Метод простой в понимании
* Устойчив, относительно ошибок округления
* Необходимо на каждом шаге вычислять f(x) и f(y) не используя внутри точки с уже вычисленным значением функции

У метода золотого сечения:

* Высокая эффективность
* Большое количество итераций

У метода Фибоначчи:

* Дает найти конкретный отрезок локализации по сравнению с другими методами
* Использует деление отрезка на 3 части в классе унимодальной функции
* Необходимость использования чисел Фибоначчи и правильно выбрать N
* Метод Фибоначчи неудобен да счет ошибок округления

# ПРИЛОЖЕНИЕ

Код для индивидуального задания 1:

**def** f(x):  
 **return** 2\*x \* x + 2 \* x + 7/2 # put your function here  
  
  
a = -3 # start (a\_0)  
b = 7 # end (b\_0)  
l = 0.2 # можно сказать что это шаг на заданом отрезке локализации в заданой точке. те +-0.2 от этой точки  
eps = 0.5 # точность  
n = 100 # limit  
  
  
**def** dichotomy(a, b, eps, l, n):  
 a\_n = a  
 b\_n = b  
 k = -1  
 **while** b\_n - a\_n > eps **or** n < 100:  
 k += 1  
 x\_n = (a\_n + b\_n - l) / 2.0  
 y\_n = (a\_n + b\_n + l) / 2.0  
 print("Step", k)  
 print("y =", x\_n, ",z =", y\_n)  
 **if** f(x\_n) < f(y\_n):  
 b\_n = y\_n  
 print(f(x\_n), "<", f(y\_n))  
 **else**:  
 a\_n = x\_n  
 print(f(x\_n), ">=", f(y\_n))  
 print("L =[", a\_n, ";", b\_n, "] and", "|L| = ", abs(a\_n-b\_n))  
 print("\nAnswer:", (a\_n + b\_n) / 2)  
 print("k =", k, ", N =", 2 \* (k + 1))  
  
  
dichotomy(a, b, eps, l, n)

Код для индивидуального задания 2:

**import** numpy  
  
  
**def** f(x):  
 **return** 2\*x \* x + 2 \* x + 7/2 # put your function here  
  
  
a = -3 # start (a\_0)  
b = 7 # end (b\_0)  
l = 0.5 # (l) - find +-  
n = 100 # limit  
  
  
**def** golden\_section(a, b, l, n):  
 a\_n = a  
 b\_n = b  
 k = 0  
 y\_n = a\_n + (3 - numpy.sqrt(5)) / 2 \* (b\_n - a\_n)  
 z\_n = a\_n + b\_n - y\_n  
 **while** b\_n - a\_n > l **or** n < 100:  
 print("Step", k)  
 print("y =", y\_n, ",z =", z\_n)  
 **if** f(y\_n) > f(z\_n):  
 print(f(y\_n), ">", f(z\_n))  
 a\_n = y\_n  
 y\_n = z\_n  
 z\_n = a\_n + b\_n - z\_n  
 **else**:  
 print(f(y\_n), "<=", f(z\_n))  
 b\_n = z\_n  
 z\_n = y\_n  
 y\_n = a\_n + b\_n - y\_n  
 print("L = [", a\_n, ";", b\_n, "]")  
 k += 1  
 print("\nAnswer:", (a\_n + b\_n) / 2)  
 print("k =", k, ", N =", 2 \* (k + 1))  
  
  
golden\_section(a, b, l, n)

Код для индивидуального задания 3:

**def** f(x):  
 **return** 2 \* x \* x + 2 \* x + 7 / 2 # put your function here  
  
**def** fib(n):  
 **if** n **in** (0, 1):  
 **return** 1  
 **return** fib(n - 1) + fib(n - 2)  
  
  
a = -3 # start (a\_0)  
b = 7 # end (b\_0)  
l = 0.5 # (l) - find +-  
eps = 0.2  
  
**def** fibonacci(a, b, l, eps):  
 n = 1  
 **while** fib(n) <= abs(b - a)/l:  
 n += 1  
 a\_n = a  
 b\_n = b  
 y\_n = a\_n + (fib(n - 2) / fib(n)) \* (b\_n - a\_n)  
 z\_n = a\_n + (fib(n - 1) / fib(n)) \* (b\_n - a\_n)  
 k = 0  
 **while** k != n - 3:  
 print("Step", k)  
 print("y =", y\_n,",z =", z\_n)  
 **if** f(y\_n) > f(z\_n):  
 print(f(y\_n), ">", f(z\_n))  
 a\_n = y\_n  
 y\_n = z\_n  
 z\_n = a\_n + (fib(n-k-2)/fib(n-k-1)) \* (b\_n - a\_n)  
 **else**:  
 print(f(y\_n), "<=", f(z\_n))  
 b\_n = z\_n  
 z\_n = y\_n  
 y\_n = a\_n + (fib(n-k-3)/fib(n-k-1)) \* (b\_n - a\_n)  
 print("L = [", a\_n, ";", b\_n, "] and", "|L| =", abs(a\_n-b\_n))  
 k += 1  
 y\_n = z\_n  
 z\_n = y\_n + eps  
 **if** f(y\_n) <= f(z\_n):  
 b\_n = z\_n  
 **else**:  
 a\_n = y\_n  
 print("\nAnswer:", (a\_n + b\_n) / 2)  
 print("k =", k)  
 string = ""  
 **for** i **in** range(n):  
 string += str(fib(i)) + ", "  
 print(string)  
  
  
fibonacci(a, b, l, eps)