**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего профессионального образования**

**«Московский государственный технический университет имени Н.Э. Баумана»**

**(МГТУ им. Н.Э.Баумана)**

**Мытищинский филиал**

**ФАКУЛЬТЕТ КОСМИЧЕСКИЙ**

**КАФЕДРА К-1 САУ**

**ЛАБОРАТОРНАЯ РАБОТА №5**

**ПО ДИСЦИПЛИНЕ**

**“Численные методы”**

**НА ТЕМУ:**

**«Решение нелинейных уравнений методом итераций»**

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

**Студент К1-61Б**  **25.04.24 Тимофеев К. А.**

(Группа) (Подпись, дата) (ФИО)

**Руководитель**  **Чернова Т.В.**

(Подпись, дата) (ФИО)

*2024 г.*

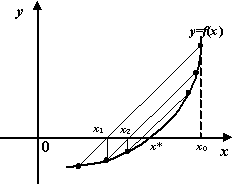
# Задачи

1. Найти корень уравнения ; методом итераций.
2. Найти второй корень уравнения ; на отрезке  методом итераций.
3. Найти корень уравнения из приложения по своему варианту (12) ; методом итераций.

# Теория

## Метод итераций





# Ход работы

## Реализация

Напишем алгоритм реализации метода итераций на языке Python

def iterationsMethod(function, start, end, epsilon=0.001):

    if function(start)\*function(end) > 0:

        print(f"Error: No root on {[start, end]}")

        return None

    x0 = (start+end)/2  # X0

    current = [x0]      # Xn

    dfadx = dfdx(function, start, epsilon)  # df(a) / dx

    dfbdx = dfdx(function, end, epsilon)    # df(b) / dx

    # C = +- 1/ max[a,b]{abs|f(x)|} = +- 1/ max{abs[|f(a)|, |f(b)|]}

    # вернёт С противоположное по знаку 1/f'(x),

    # где X - либо start, если |f'(start)| > |f'(end)|,

    #         либо end, если |f'(end)| > |f'(start)|

    max = np.max([abs(dfadx), abs(dfbdx)])

    if dfadx > 0 or dfbdx > 0 : c = 1 / - max

    else: c = - 1 / - max

print(f"C = {c}")

new = [x0 + c \* function(x0)]    # Xn+1

    f = [function(new[len(new)-1])]  # F(Xn+1)

    epsControl = [new[len(new)-1] - current[len(current)-1]] # оценка погрешности

    it = 1  # номер итерации

    # цикл итераций

    while abs((new[len(new)-1] - current[len(current)-1])) > epsilon:

        if  new[len(new)-1] > end:

            print("No root")

            return pd.DataFrame({

        "Xn": current, "Xn+1": new,

        "Оценка погрешности": epsControl, " Контроль нуля Y(Xn+1)": f,

        "№": [i for i in range(1, it+1)]

    })

        current.append(new[len(new)-1])

        new.append(current[len(current)-1] + c \* function(current[len(current)-1]))

        f.append(function(new[len(new)-1]))

        epsControl.append(abs(new[len(new)-1] - current[len(current)-1]))

        it+=1

    epsControl.pop()

    epsControl.append(f"Корень = {round(new[len(new)-1], 4)}")

    return pd.DataFrame({ ···

Вспомогательные функции

def f(x):

    return x\*\*2 - np.sqrt(x+4)

def f1(x):

return x\*\*3 + 3\*x\*\*2 -1

def dfdx(function, x, epsilon):

return 0.5\*(function(x + epsilon)-function(x - epsilon))/epsilon

def d2fdx2(function, x, epsilon):

    return (function(x-epsilon)-2\*function(x)+function(x+epsilon))/epsilon\*\*2

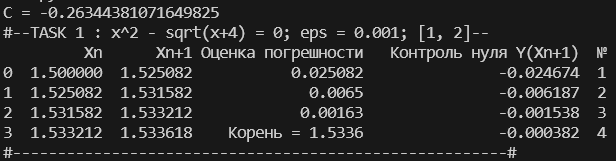
## Задание 1

# TASK 1 : x^2 - sqrt(x+4) = 0; eps = 0.001; [1, 2]

print( f"#--TASK 1 : x^2 - sqrt(x+4) = 0; eps = 0.001; [1, 2]--\n

{iterationsMethod(f, 1.0, 2.0, 0.001)} \

      \n#-------------------------------------------------------#")



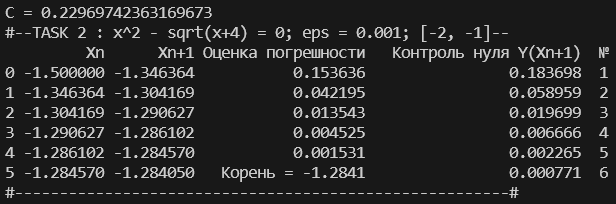
## Задание 2

# TASK 2 : x^2 - sqrt(x+4) = 0; eps = 0.001; [-2, -1]

print( f"#--TASK 2 : x^2 - sqrt(x+4) = 0; eps = 0.001; [-2, -1]--\n

{iterationsMethod(f, -2.0, -1.0, 0.001)} \

      \n#-------------------------------------------------------#")



## Задание 3

# TASK 3 : x\*\*3 + 3\*x\*\*2 -1 = 0; eps = 0.001; [-3, -2] & [-1, 0] & [0, 1]

print( f"#--TASK 3 : x\*\*3 + 3\*x\*\*2 -1 = 0; eps = 0.001; [-3, -2]--\n

{iterationsMethod(f1, -3.0, -1.0, 0.001)} \

\n#-------------------------------------------------------#")

