**NC ASSIGNMENT 01**

**Name: Bilal Ahmed Khan Sec: B Roll No: 20k-0183**

**Taylor’s polynomial**

print("This is Taylor Polynomial File")

import math

import numpy as np

import matplotlib.pyplot as plt

def func\_sin(x, n):

    sin\_approx = 0

    for i in range(n):

        coef = (-1)\*\*(i)

        num = x\*\*((2\*i)+1)

        denom = math.factorial((2\*i)+1)

        sin\_approx += ( coef ) \* ( (num)/(denom) )

    return sin\_approx

angles = np.arange(-2\*np.pi,2\*np.pi,0.1)

p\_sin = np.sin(angles)

t\_sin = [func\_sin(angle,5) for angle in angles]

fig, ax = plt.subplots()

ax.plot(angles,p\_sin)

ax.plot(angles,t\_sin)

ax.set\_ylim([-5,5])

plt.xlabel("Value of Angles (in radians)")

plt.ylabel("Value of sin()")

plt.title("Value of sin() approximated using Taylor's polynomial\n as compared to the actual function", loc= 'center')

plt.show()

**Bisection Method**

import math

import numpy as np

import matplotlib.pyplot as plt

def f(x):

    return x\*\*3-5\*x-9

def bisection(x0,x1,e):

    step = 1

    condition = True

    while condition:

        x2 = (x0 + x1)/2

        print('c = %0.6f and f(c) = %0.6f' % ( x2, f(x2)))

        if f(x0) \* f(x2) < 0:

            x1 = x2

        else:

            x0 = x2

        step = step + 1

        condition = abs(f(x2)) > e

    print('\nRoot is : %0.8f' % x2)

# Converting input to float

x0 = -100

x1 = 100

e = float(math.e)

if f(x0) \* f(x1) > 0.0:

    print('Given guess values do not bracket the root.')

    print('Try Again with different guess values.')

else:

    bisection(x0,x1,e)

**Fixed Point**

import math

def f(x):

    return x\*x\*x + x\*x -1

def g(x):

    return 1/math.sqrt(1+x)

def fixedPointIteration(x0, e, N):

    print('\n\n\*\*\* FIXED POINT ITERATION \*\*\*')

    step = 1

    flag = 1

    condition = True

    while condition:

        x1 = g(x0)

        print('Iteration-%d, x1 = %0.6f and f(x1) = %0.6f' % (step, x1, f(x1)))

        x0 = x1

        step = step + 1

        if step > N:

            flag=0

            break

        condition = abs(f(x1)) > e

    if flag==1:

        print('\nRequired root is: %0.8f' % x1)

    else:

        print('\nNot Convergent.')

x0 = input('Enter Guess: ')

e = input('Tolerable Error: ')

N = input('Maximum Step: ')

x0 = float(x0)

e = float(e)

N = int(N)

fixedPointIteration(x0,e,N)

Lagrange Interpolation

import numpy as np

n = int(input('Enter number of data points: '))

x = np.zeros((n))

y = np.zeros((n))

print('Enter data for x and y: ')

for i in range(n):

    x[i] = float(input( 'x['+str(i)+']='))

    y[i] = float(input( 'y['+str(i)+']='))

xp = float(input('Enter interpolation point: '))

yp = 0

for i in range(n):

    p = 1

    for j in range(n):

        if i != j:

            p = p \* (xp - x[j])/(x[i] - x[j])

    yp = yp + p \* y[i]

print('Interpolated value at %.3f is %.3f.' % (xp, yp))