Звіт до лабораторної роботи №8

з дисципліни «Чисельні методи програмування»

студентки 2 курсу 6 групи ФІТ

Маргаза Дар’ї Юріївни

**Тема:** «Інтерполяційний багаточлен Ньютона».

**Хід виконання роботи**

Наближено відбудувати функцію , що задана таблицею, у довільній точці х за допомогою інтерполяційних багаточленів Ньютона. За наявним набором значень побудувати графік інтерполяційної функції.

**Варіант 16**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | № варіанта | Значення аргумента х |
| 0,115  0,120  0,125  0,130  0,135  0,140  0,145  0,150  0,155  0,160  0,165 | 8,6572  8,2932  7,9582  7,6489  7,3623  7,0961  6,8491  6,6185  6,3998  6,1965  6,0055 | 16 | 0,129 … 0,157 |

**Код:**

import numpy as np

from math import factorial

import matplotlib.pyplot as plt

x = [0.115, 0.120, 0.125, 0.130, 0.135, 0.140]

y = [8.6572, 8.2932, 7.9582, 7.6489, 7.3623, 7.0961]

h = x[1] - x[0]

x1 = 0.129

x2 = 0.157

q1 = (x1 - x[0]) / h

q2 = (x2 - x[-1]) / h

def finite\_differences(y, j):

mas = []

for i in range(len(y)):

mas.append(y[i] - y[i - 1])

mas.pop(0)

if j == 1:

return mas

else:

j -= 1

return finite\_differences(mas, j)

#Newton's first interpolation formula

s1 = y[0] + q1 \* finite\_differences(y, 1)[0] + q1 \* (q1 - 1) \* finite\_differences(y, 2)[0] / factorial(2)

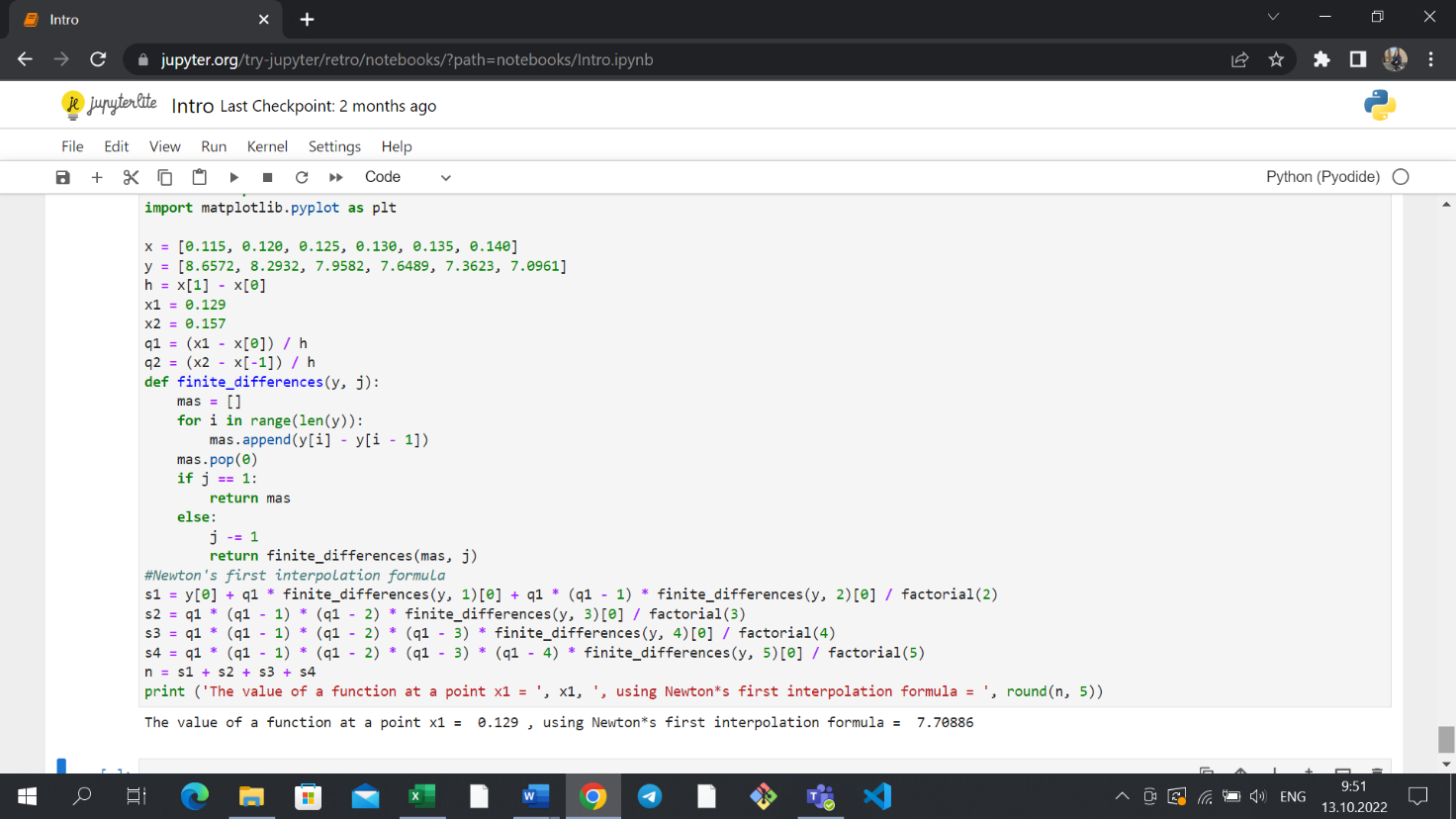
s2 = q1 \* (q1 - 1) \* (q1 - 2) \* finite\_differences(y, 3)[0] / factorial(3)

s3 = q1 \* (q1 - 1) \* (q1 - 2) \* (q1 - 3) \* finite\_differences(y, 4)[0] / factorial(4)

s4 = q1 \* (q1 - 1) \* (q1 - 2) \* (q1 - 3) \* (q1 - 4) \* finite\_differences(y, 5)[0] / factorial(5)

n = s1 + s2 + s3 + s4

print ('The value of a function at a point x1 = ', x1, ', using Newton\*s first interpolation formula = ', round(n, 5))



#Newton's second interpolation formula

m1 = y[5] + q2 \* finite\_differences(y, 1)[4] \* (q2 \* (q2 + 1) / factorial(2)) \* finite\_differences(y, 2)[3]

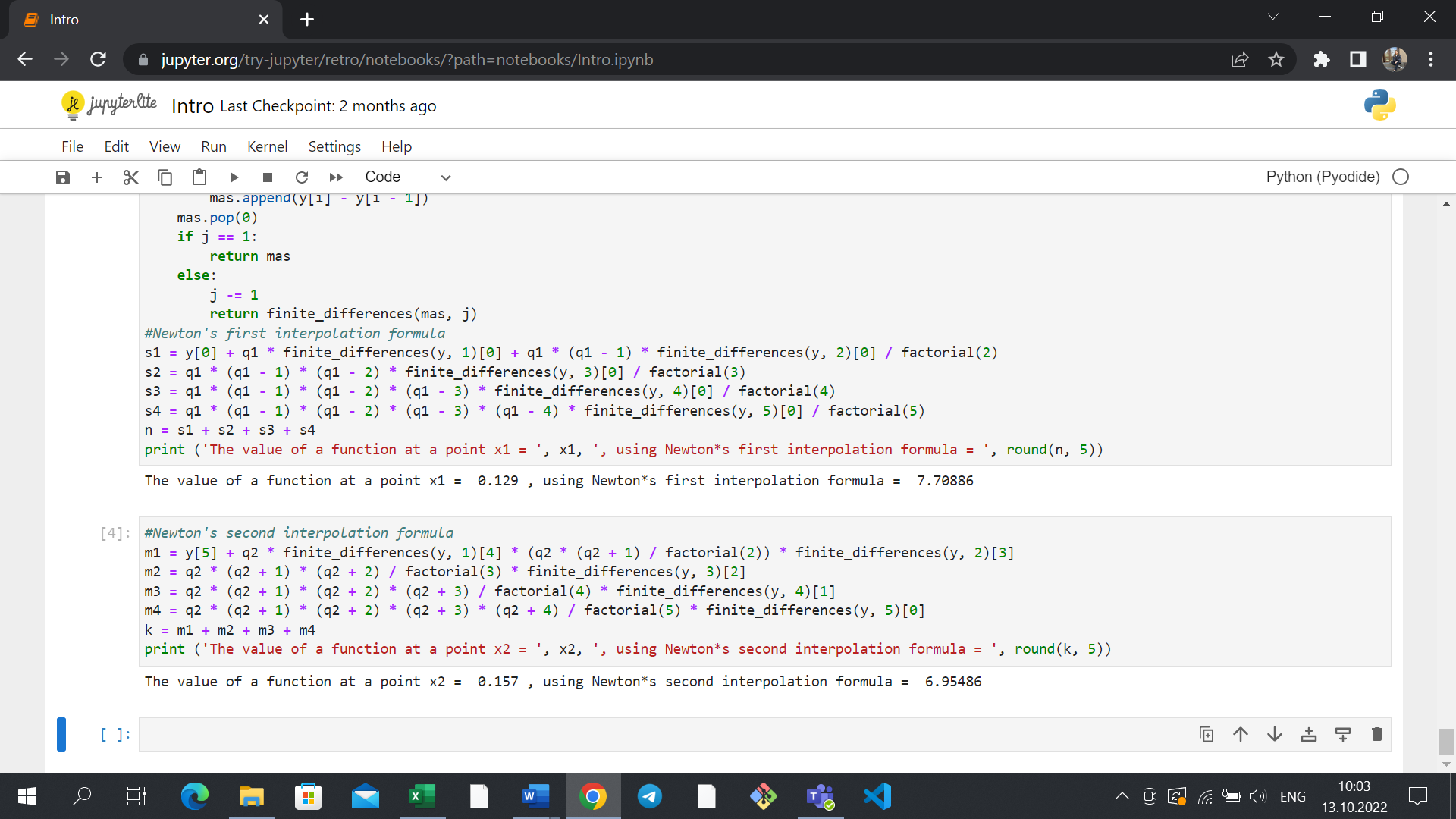
m2 = q2 \* (q2 + 1) \* (q2 + 2) / factorial(3) \* finite\_differences(y, 3)[2]

m3 = q2 \* (q2 + 1) \* (q2 + 2) \* (q2 + 3) / factorial(4) \* finite\_differences(y, 4)[1]

m4 = q2 \* (q2 + 1) \* (q2 + 2) \* (q2 + 3) \* (q2 + 4) / factorial(5) \* finite\_differences(y, 5)[0]

k = m1 + m2 + m3 + m4

print ('The value of a function at a point x2 = ', x2, ', using Newton\*s second interpolation formula = ', round(k, 5))



x\_1 = np.linspace(np.min(x), np.max(x))

y\_1 = np.interp(x\_1, x, y)

plt.plot(x, y, 'o', x\_1, y\_1)

plt.title('Graph of the interpolation function')

plt.xlabel('x')

plt.ylabel('y')

plt.grid()

plt.show()

