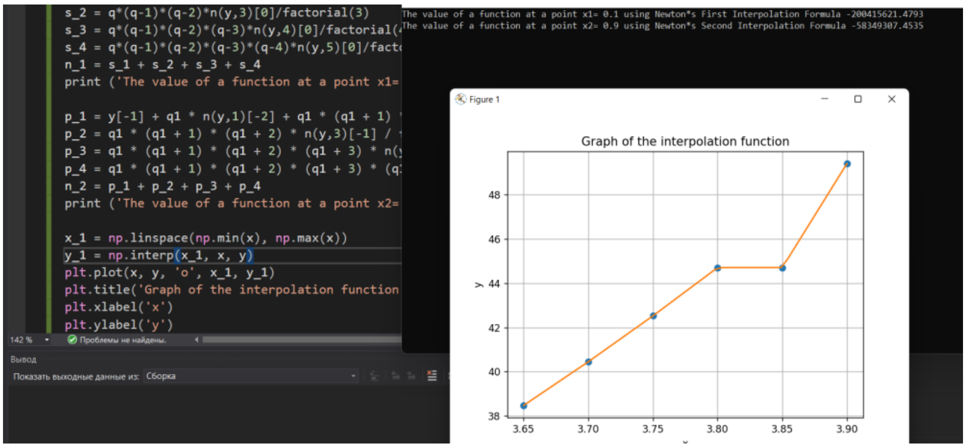


```
import numpy as np

from math import factorial

import matplotlib.pyplot as plt


x=[3.65, 3.70, 3.75, 3.80, 3.85, 3.90]

y=[38.4747, 40.4473, 42.5211, 44.7012, 44.7012, 49.4024]

h = x[1] - x[0]


x1=0.1

x2=0.9


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

q1 = (x2-x[-1])/h


def n(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 n(mas, j)

s_1 = y[0]+q*n(y,1)[0]+q*(q-1)*n(y,2)[0]/factorial(2)
s_2 = q*(q-1)*(q-2)*n(y,3)[0]/factorial(3)
s_3 = q*(q-1)*(q-2)*(q-3)*n(y,4)[0]/factorial(4)
s_4 = q*(q-1)*(q-2)*(q-3)*(q-4)*n(y,5)[0]/factorial(5)
n_1 = s_1 + s_2 + s_3 + s_4
print ('The value of a function at a point x1=', x1, 'using Newton*s First Interpolation Formula', round(n_1,5))

p_1 = y[-1] + q1 * n(y,1)[-2] + q1 * (q1 + 1) * n(y,2)[-3] / factorial(2)
p_2 = q1 * (q1 + 1) * (q1 + 2) * n(y,3)[-1] / factorial(3)
p_3 = q1 * (q1 + 1) * (q1 + 2) * (q1 + 3) * n(y,4)[-1] / factorial(4)
p_4 = q1 * (q1 + 1) * (q1 + 2) * (q1 + 3) * (q1 + 4) * n(y,5)[-1] / factorial(5)
n_2 = p_1 + p_2 + p_3 + p_4
print ('The value of a function at a point x2=', x2, 'using Newton*s Second Interpolation Formula', round(n_2,5))

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()
```