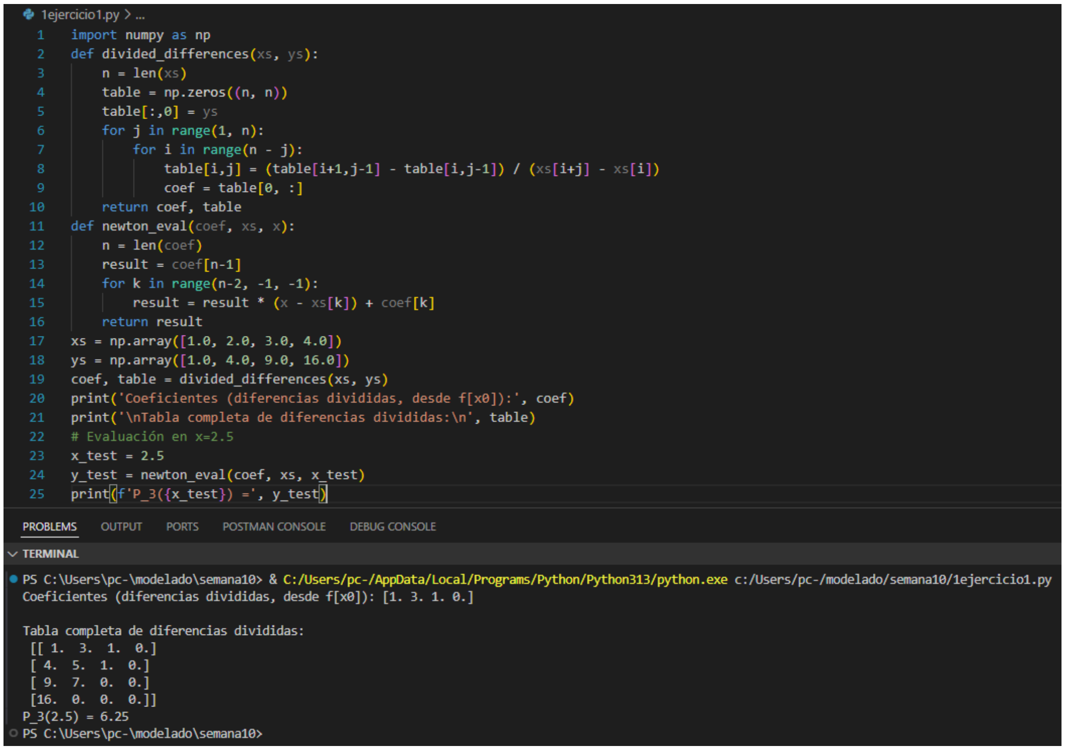
Actividad 2 – II Unidad: Interpolación de Newton

1- EJERCICIO

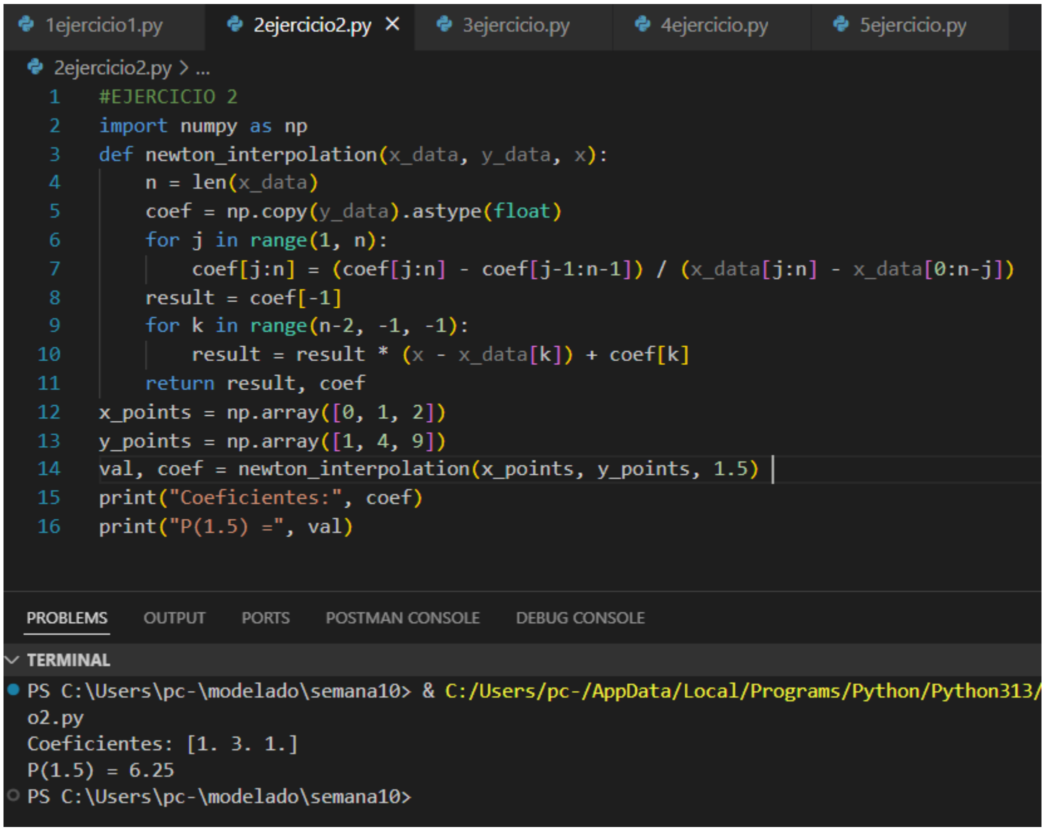


```python
import numpy as np
def divided_differences(xs, ys):
    n = len(xs)
    table = np.zeros((n, n))
    table[:,0] = ys
    for j in range(1, n):
        for i in range(n - j):
            table[i,j] = (table[i+1,j-1] - table[i,j-1]) / (xs[i+j] - xs[i])
            coef = table[0, :]
    return coef, table
def newton_eval(coef, xs, x):
    n = len(coef)
    result = coef[n-1]
    for k in range(n-2, -1, -1):
        result = result * (x - xs[k]) + coef[k]
    return result
xs = np.array([1.0, 2.0, 3.0, 4.0])
ys = np.array([1.0, 4.0, 9.0, 16.0])
coef, table = divided_differences(xs, ys)
print('Coeficientes (diferencias divididas, desde f[x0]):', coef)
print('\nTabla completa de diferencias divididas:\n', table)
# Evaluación en x=2.5
x_test = 2.5
y_test = newton_eval(coef, xs, x_test)
print(f'P_3({x_test}) =', y_test)
```

```
PS C:\Users\pc-\modelado\semana10> & C:/Users/pc-/AppData/Local/Programs/Python/Python313/python.exe c:/Users/pc-/modelado/semana10/1ejercicio1.py
Coeficientes (diferencias divididas, desde f[x0]): [1. 3. 1. 0.]

Tabla completa de diferencias divididas:
 [[ 1.  3.  1.  0.]
 [ 4.  5.  1.  0.]
 [ 9.  7.  0.  0.]
 [16.  0.  0.  0.]]
P_3(2.5) = 6.25
PS C:\Users\pc-\modelado\semana10>
```

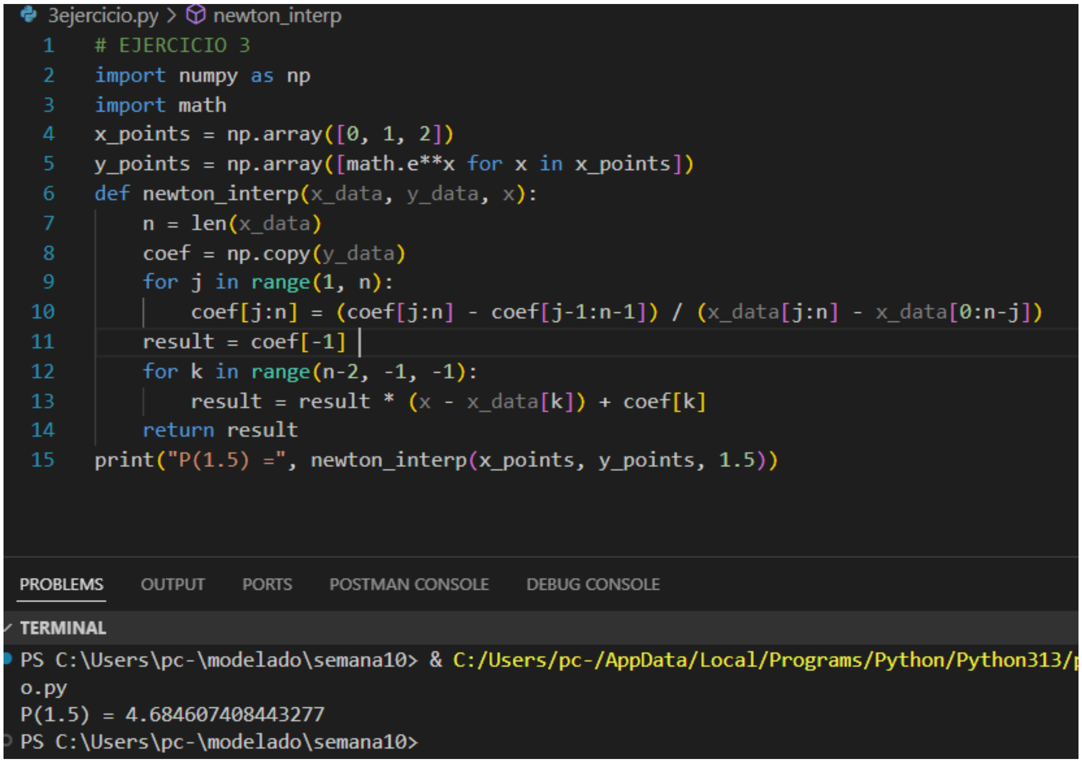
2- EJERCICIO



```python
#EJERCICIO 2
import numpy as np
def newton_interpolation(x_data, y_data, x):
    n = len(x_data)
    coef = np.copy(y_data).astype(float)
    for j in range(1, n):
        coef[j:n] = (coef[j:n] - coef[j-1:n-1]) / (x_data[j:n] - x_data[0:n-j])
    result = coef[-1]
    for k in range(n-2, -1, -1):
        result = result * (x - x_data[k]) + coef[k]
    return result, coef
x_points = np.array([0, 1, 2])
y_points = np.array([1, 4, 9])
val, coef = newton_interpolation(x_points, y_points, 1.5)
print("Coeficientes:", coef)
print("P(1.5) =", val)
```

```
PS C:\Users\pc-\modelado\semana10> & C:/Users/pc-/AppData/Local/Programs/Python/Python313/
o2.py
Coeficientes: [1. 3. 1.]
P(1.5) = 6.25
PS C:\Users\pc-\modelado\semana10>
```

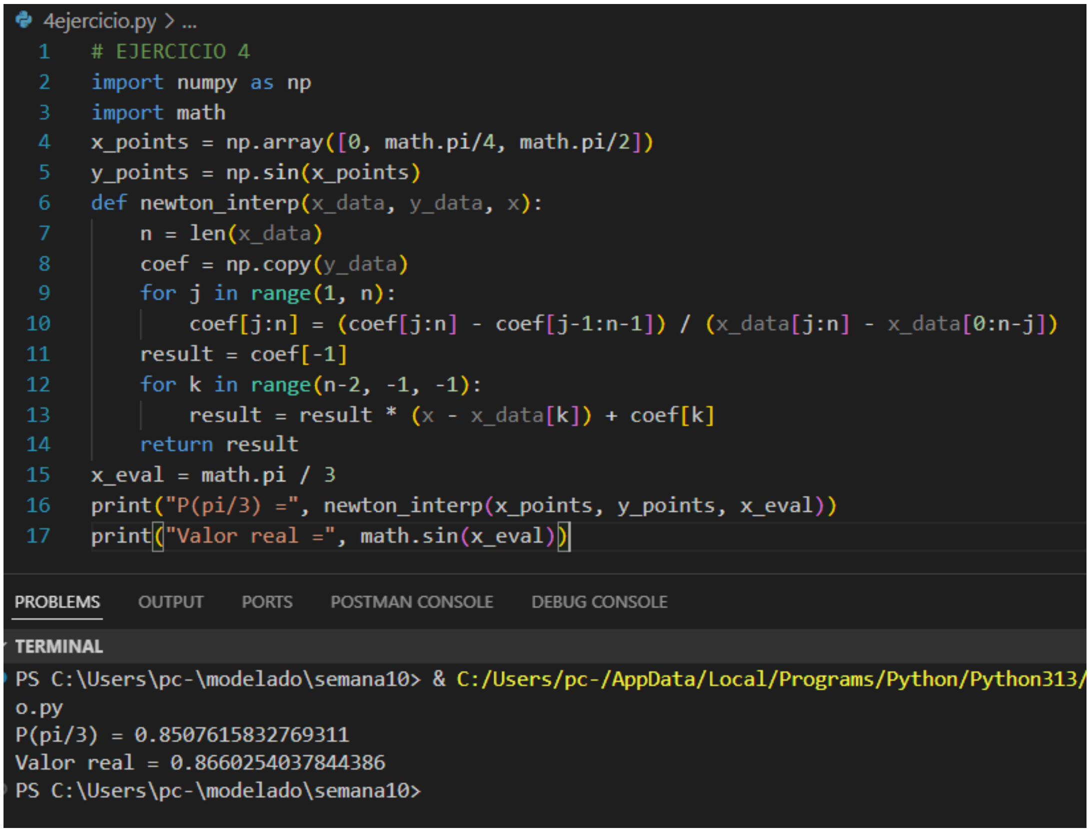
3- EJERCICIO



```python
# EJERCICIO 3
import numpy as np
import math
x_points = np.array([0, 1, 2])
y_points = np.array([math.e**x for x in x_points])
def newton_interp(x_data, y_data, x):
    n = len(x_data)
    coef = np.copy(y_data)
    for j in range(1, n):
        coef[j:n] = (coef[j:n] - coef[j-1:n-1]) / (x_data[j:n] - x_data[0:n-j])
    result = coef[-1]
    for k in range(n-2, -1, -1):
        result = result * (x - x_data[k]) + coef[k]
    return result
print("P(1.5) =", newton_interp(x_points, y_points, 1.5))
```

```
PS C:\Users\pc-\modelado\semana10> & C:/Users/pc-/AppData/Local/Programs/Python/Python313/
o.py
P(1.5) = 4.684607408443277
PS C:\Users\pc-\modelado\semana10>
```

4- EJERCICIO



```python
# EJERCICIO 4
import numpy as np
import math
x_points = np.array([0, math.pi/4, math.pi/2])
y_points = np.sin(x_points)
def newton_interp(x_data, y_data, x):
    n = len(x_data)
    coef = np.copy(y_data)
    for j in range(1, n):
        coef[j:n] = (coef[j:n] - coef[j-1:n-1]) / (x_data[j:n] - x_data[0:n-j])
    result = coef[-1]
    for k in range(n-2, -1, -1):
        result = result * (x - x_data[k]) + coef[k]
    return result
x_eval = math.pi / 3
print("P(pi/3) =", newton_interp(x_points, y_points, x_eval))
print("Valor real =", math.sin(x_eval))
```

```
PS C:\Users\pc-\modelado\semana10> & C:/Users/pc-/AppData/Local/Programs/Python/Python313/
o.py
P(pi/3) = 0.8507615832769311
Valor real = 0.8660254037844386
PS C:\Users\pc-\modelado\semana10>
```