

Tarefa 1 - Unidad 1

Python - serie de Fibonacci

Posta en producción segura 2023-2024. (Grupo A)

Alumno: *Juan Pablo Bongiovane Amado*

Docente: *José Fernández Gómez*

Tarefa 1 - Unidad 1

Posta en producción segura 2023-2024. (Grupo A)

Ejercicio

Función `fibo.py`

Programa principal `main`

Resultados

Capturas del código

Bibliografía

Ejercicio

Para la resolución del ejercicio se utilizara el lenguaje de programación Python.

Serie de Fibonacci: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34...

Función `fibonacci.py`

Creamos un script llamado `fibonacci.py` donde dentro se define una función llamada `fibonacci`.

```
def fibonacci(n):  
  
    a = 0  
    b = 1  
  
    if n<=1:  
        return n  
  
    for i in range(n-1):  
        temp = a + b  
        a = b  
        b = temp  
    return a
```

La función calcula el número de fibonacci en la posición `n` y devuelve el valor del numero.

Programa principal `main`

Primero importamos la librería para realizar el testeo de la función `unittest` y la función de `fibonacci()` creada en `fibonacci.py`.

Creamos una clase del tipo `unittest.TestCase` y en su interior definimos `test_fibonacci`, donde verificamos si la posición y devuelve el valor esperado de la sucesión.

Se generan tres casos, de los cuales uno se espera que no supere la prueba, con el objetivo de evaluar el comportamiento del programa ante un test fallido.

```
import unittest #import de la libreria de testeo  
from fibo import fibonacci #import función fibonacci  
  
class TestFibonacci(unittest.TestCase):  
    def __init__(self, methodName, param1=None, param2=None):  
        super(TestFibonacci, self).__init__(methodName)  
  
        self.param1 = param1  
        self.param2 = param2  
  
    def test_fibonacci(self):  
        resultado = fibonacci(self.param1)  
        self.assertEqual(resultado, self.param2)
```

```
if __name__ == "__main__":
    test_cases = unittest.TestSuite()
    test_cases.addTest(TestFibonacci('test_fibonacci', 5, 3))
    test_cases.addTest(TestFibonacci('test_fibonacci', 5, 0))
    test_cases.addTest(TestFibonacci('test_fibonacci', 2, 1))
    unittest.TextTestRunner().run(test_cases)
```

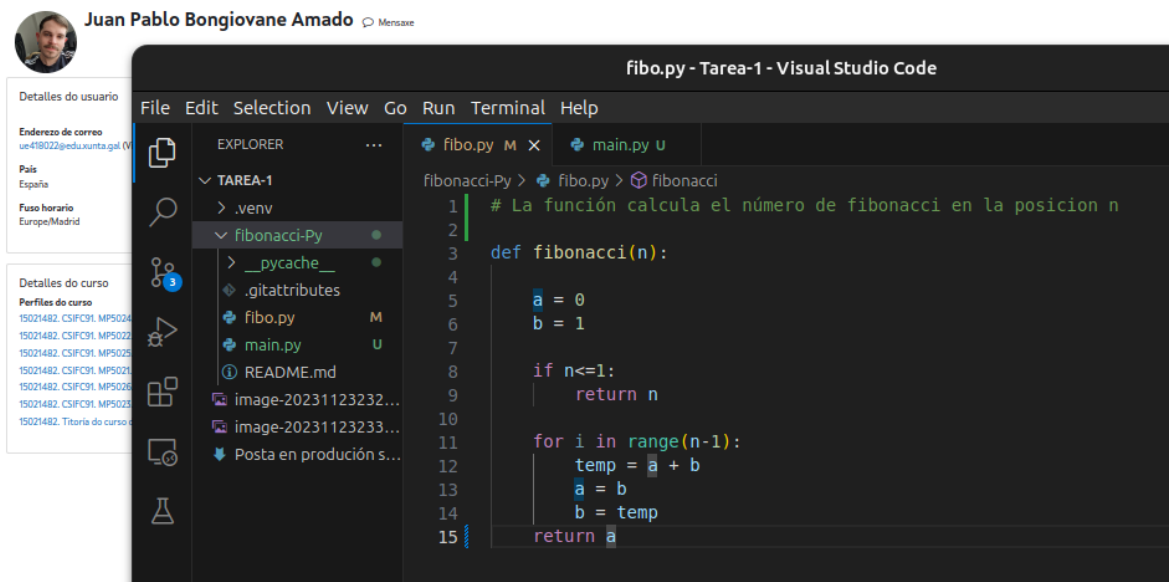
Resultados


Al ejecutar el programa el test verificara si el programa que calcula la sucesión de fibonacci se comporta como espera.

```
(.venv) juan@juan-USB:~/Escritorio/Ciberseguridad/Posta en Producción Segura/Tarea-1$ "/home/juan/Escritorio/Ciberseguridad/Posta en Producción Segura/Tarea-1/.venv/bin/python" "/home/juan/Escritorio/Ciberseguridad/Posta en Producción Segura/Tarea-1/fibonacci-Py/main.py"
.F.
=====
FAIL: test_fibonacci (__main__.TestFibonacci.test_fibonacci)
-----
Traceback (most recent call last):
  File "/home/juan/Escritorio/Ciberseguridad/Posta en Producción Segura/Tarea-1/fibonacci-Py/main.py", line 15, in test_fibonacci
    self.assertAlmostEqual(resultado, self.param2)
AssertionError: 3 != 0 within 7 places (3 difference)
-----
Ran 3 tests in 0.000s

FAILED (failures=1)
(.venv) juan@juan-USB:~/Escritorio/Ciberseguridad/Posta en Producción Segura/Tarea-1$
```

Capturas del código





Juan Pablo Bongiovane Amado Mensaxe

Detalles do usuario

Enderezo de correo

ue418022@edu.xunta.gal (V)

País

España

Fuso horario

Europe/Madrid

Detalles do curso

Perfiles do curso

15021482. CSIFC91. MP5024

15021482. CSIFC91. MP5022

15021482. CSIFC91. MP5025

15021482. CSIFC91. MP5021

15021482. CSIFC91. MP5026

15021482. CSIFC91. MP5023

15021482. Titoría do curso

2024. (Grupo A)

main.py - Tarea-1 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

TAREA-1

.venv

fibonacci-Py

__pycache__

.gitattributes

fibonacci.py

main.py

Posta en prod...

README.md

image-20231123232...

image-20231123233...

image-20231123233...

image-20231123233...

image-20231124002...

Posta en produción s...

fibonacci.py

main.py

1 import unittest #import de la libreria de testeo

2 from fibo import fibonacci #import función fibonacci

3

4

5 class TestFibonacci(unittest.TestCase):

6 def __init__(self, methodName, param1=None, param2=None):

7 super(TestFibonacci, self).__init__(methodName)

8

9 self.param1 = param1

10 self.param2 = param2

11

12

13

14 def test_fibonacci(self):

15 resultado = fibonacci(self.param1)

16 self.assertEqual(resultado, self.param2)

17

18

19 if __name__ == "__main__":

20 test_cases = unittest.TestSuite()

21 test_cases.addTest(TestFibonacci('test_fibonacci', 5, 3))

22 test_cases.addTest(TestFibonacci('test_fibonacci', 5, 0))

23 test_cases.addTest(TestFibonacci('test_fibonacci', 2, 1))

24 unittest.TextTestRunner().run(test_cases)

Bibliografía

- [documentación de Python - 3.12.0](#)
- [unittest — Infraestructura de tests unitarios](#)
- [Sucesión de Fibonacci](#)