

# Posta en Producción Segura

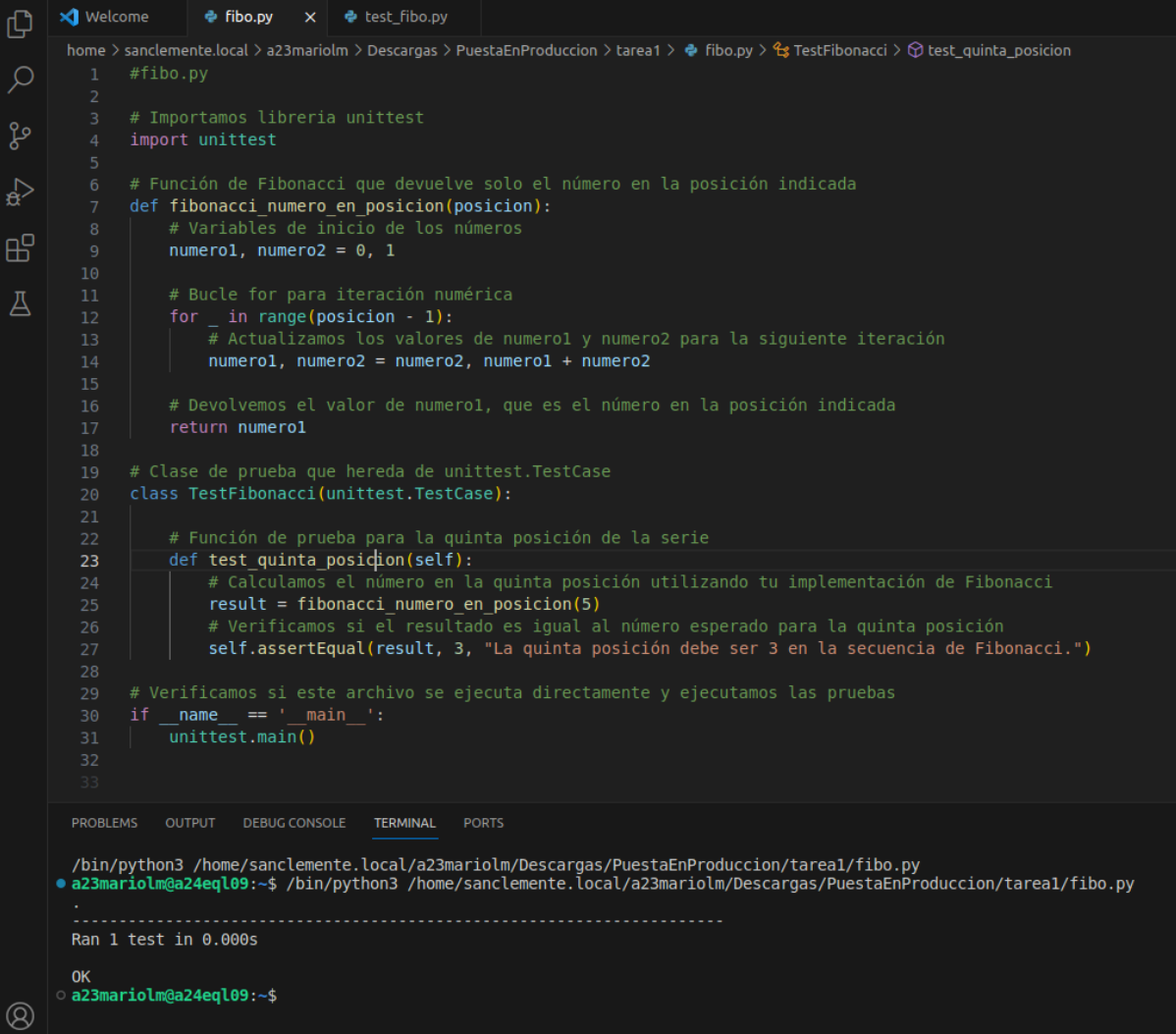
## Tarefa 1

Mario Loureiro Montenegro

<b>Código entrega:</b> .....	<b>3</b>
<b>Estructura del código:</b> .....	<b>4</b>
Importación de librería "unittest":.....	4
Definición de la Función de Fibonacci:.....	4
Definición de la Clase de Prueba:.....	5
Ejecución de las Pruebas:.....	5
<b>Funcionamiento general:</b> .....	<b>6</b>
<b>Bibliografía:</b> .....	<b>6</b>
<b>Anexos:</b> .....	<b>6</b>
Código completo de pruebas:.....	6
Código completo final (entrega):.....	7

## Código entrega:

En la siguiente imagen se muestra todo el código utilizado para la práctica. En el siguiente apartado explicaré cada una de las líneas y su función.



```

1 #fibonacci.py
2
3 # Importamos libreria unittest
4 import unittest
5
6 # Función de Fibonacci que devuelve solo el número en la posición indicada
7 def fibonacci_numero_en_posicion(posicion):
8     # Variables de inicio de los números
9     numero1, numero2 = 0, 1
10
11     # Bucle for para iteración numérica
12     for _ in range(posicion - 1):
13         # Actualizamos los valores de numero1 y numero2 para la siguiente iteración
14         numero1, numero2 = numero2, numero1 + numero2
15
16     # Devolvemos el valor de numero1, que es el número en la posición indicada
17     return numero1
18
19 # Clase de prueba que hereda de unittest.TestCase
20 class TestFibonacci(unittest.TestCase):
21
22     # Función de prueba para la quinta posición de la serie
23     def test_quinta_posicion(self):
24         # Calculamos el número en la quinta posición utilizando tu implementación de Fibonacci
25         result = fibonacci_numero_en_posicion(5)
26         # Verificamos si el resultado es igual al número esperado para la quinta posición
27         self.assertEqual(result, 3, "La quinta posición debe ser 3 en la secuencia de Fibonacci.")
28
29 # Verificamos si este archivo se ejecuta directamente y ejecutamos las pruebas
30 if __name__ == '__main__':
31     unittest.main()
32
33

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

/bin/python3 /home/sanclemente.local/a23mariolm/Descargas/PuestaEnProduccion/tarea1/fibo.py
● a23mariolm@a24eq109:~$ /bin/python3 /home/sanclemente.local/a23mariolm/Descargas/PuestaEnProduccion/tarea1/fibo.py
.
-----
Ran 1 test in 0.000s

OK
○ a23mariolm@a24eq109:~$

```

## Estructura del código:

### Importación de librería “unittest”:

```
2
3 # Importamos libreria unittest
4 import unittest
5
```

Importa la librería unittest, que proporciona un marco para escribir y ejecutar pruebas unitarias en Python.

### Definición de la Función de Fibonacci:

```
6 # Función de Fibonacci que devuelve solo el número en la posición indicada
7 def fibonacci_numero_en_posicion(posicion):
```

Define una función llamada “fibonacci\_numero\_en\_posicion” que toma un argumento “posicion”, que representa la posición en la secuencia de Fibonacci.

```
8     # Variables de inicio de los números
9     numero1, numero2 = 0, 1
10
```

Inicializa dos variables, “numero1” y “numero2”, con los primeros dos números de la secuencia de Fibonacci.

```
11     # Bucle for para iteración numérica
12     for _ in range(posicion - 1):
```

Inicia un bucle for que se ejecutará “posicion” - 1 veces, ya que ya hemos inicializado dos números.

```
13         # Actualizamos los valores de numero1 y numero2 para la siguiente iteración
14         numero1, numero2 = numero2, numero1 + numero2
15
```

En cada iteración, actualiza los valores de “numero1” y “numero2” para generar la secuencia de Fibonacci.

```
16     # Devolvemos el valor de numero1, que es el número en la posición indicada
17     return numero1
18
```

Devuelve el valor de “numero1”, que es el número en la posición indicada.

## Definición de la Clase de Prueba:

```
18
19 # Clase de prueba que hereda de unittest.TestCase
20 class TestFibonacci(unittest.TestCase):
21
```

Define una clase llamada “TestFibonacci” que hereda de `unittest.TestCase`, lo que significa que contiene pruebas unitarias.

```
21
22 # Función de prueba para la quinta posición de la serie
23 def test_quinta_posicion(self):
```

Define una función de prueba llamada “test\_quinta\_posicion” para verificar el comportamiento de la función en la quinta posición.

```
24 # Calculamos el número en la quinta posición utilizando tu implementación de Fibonacci
25 result = fibonacci_numero_en_posicion(5)
```

Calcula el resultado de la función “fibonacci\_numero\_en\_posicion” para la quinta posición.

```
26 # Verificamos si el resultado es igual al número esperado para la quinta posición
27 self.assertEqual(result, 3, "La quinta posición debe ser 3 en la secuencia de Fibonacci.")
28
```

Utiliza el método “assertEqual” de la clase “TestCase” para verificar si el resultado es igual a 3. Si no es igual, muestra un mensaje de error.

## Ejecución de las Pruebas:

```
28
29 # Verificamos si este archivo se ejecuta directamente y ejecutamos las pruebas
30 if __name__ == '__main__':
31     unittest.main()
32
```

Verifica si el script se está ejecutando directamente y, en ese caso, ejecuta las pruebas definidas en la clase “TestFibonacci” utilizando “`unittest.main()`”.

## Funcionamiento general:

La función “fibonacci\_numero\_en\_posicion” genera la secuencia de Fibonacci hasta la posición indicada y devuelve el número en esa posición.

La clase de prueba “TestFibonacci” verifica que el resultado de la función para la quinta posición sea igual a 3. Si no es así, muestra un mensaje de error.

Cuando el script se ejecuta, las pruebas definidas en la clase “TestFibonacci” se ejecutan automáticamente.

## Bibliografía:

<https://docs.python.org/3/library/unittest.html>

<https://stackoverflow.com/>

<https://github.com/>

## Anexos:

### Código completo de pruebas:

```
#test_fibo.py

#####
###parte 2###
#####

# Importamos libreria unittest
import unittest

#####
###parte 1###
#####

# Función de Fibonacci que devuelve solo el número en la posición
indicada
def fibonacci_numero_en_posicion(posicion):
    # Variables de inicio de los números
    numero1, numero2 = 0, 1

    # Bucle for para iteración numérica
```

```

    for _ in range(posicion - 1):
        # Actualizamos los valores de numero1 y numero2 para la
siguiente iteración
        numero1, numero2 = numero2, numero1 + numero2

    # Devolvemos el valor de numero1, que es el número en la posición
indicada
    #print(numero1)
    return numero1

#fibonacci_numero_en_posicion(5)

#####
###parte 3###
#####

# Clase de prueba que hereda de unittest.TestCase
class TestFibonacci(unittest.TestCase):

    # Función de prueba para la quinta posición de la serie
    def test_quinta_posicion(self):
        # Calculamos el número en la quinta posición utilizando tu
implementación de Fibonacci
        result = fibonacci_numero_en_posicion(5)
        # Verificamos si el resultado es igual al número esperado para
la quinta posición
        self.assertEqual(result, 3, "La quinta posición debe ser 3 en la
secuencia de Fibonacci.")

#####
###parte 4###
#####

# Verificamos si este archivo se ejecuta directamente y ejecutamos las
pruebas
if __name__ == '__main__':
    unittest.main()

```

## Código completo final (entrega):

```
#fibo.py

# Importamos libreria unittest
import unittest

# Función de Fibonacci que devuelve solo el número en la posición
indicada
def fibonacci_numero_en_posicion(posicion):
    # Variables de inicio de los números
    numero1, numero2 = 0, 1

    # Bucle for para iteración numérica
    for _ in range(posicion - 1):
        # Actualizamos los valores de numero1 y numero2 para la
        siguiente iteración
        numero1, numero2 = numero2, numero1 + numero2

    # Devolvemos el valor de numero1, que es el número en la posición
    indicada
    return numero1

# Clase de prueba que hereda de unittest.TestCase
class TestFibonacci(unittest.TestCase):

    # Función de prueba para la quinta posición de la serie
    def test_quinta_posicion(self):
        # Calculamos el número en la quinta posición utilizando tu
        implementación de Fibonacci
        result = fibonacci_numero_en_posicion(5)
        # Verificamos si el resultado es igual al número esperado para
        la quinta posición
        self.assertEqual(result, 3, "La quinta posición debe ser 3 en la
        secuencia de Fibonacci.")

# Verificamos si este archivo se ejecuta directamente y ejecutamos las
pruebas
if __name__ == '__main__':
    unittest.main()
```