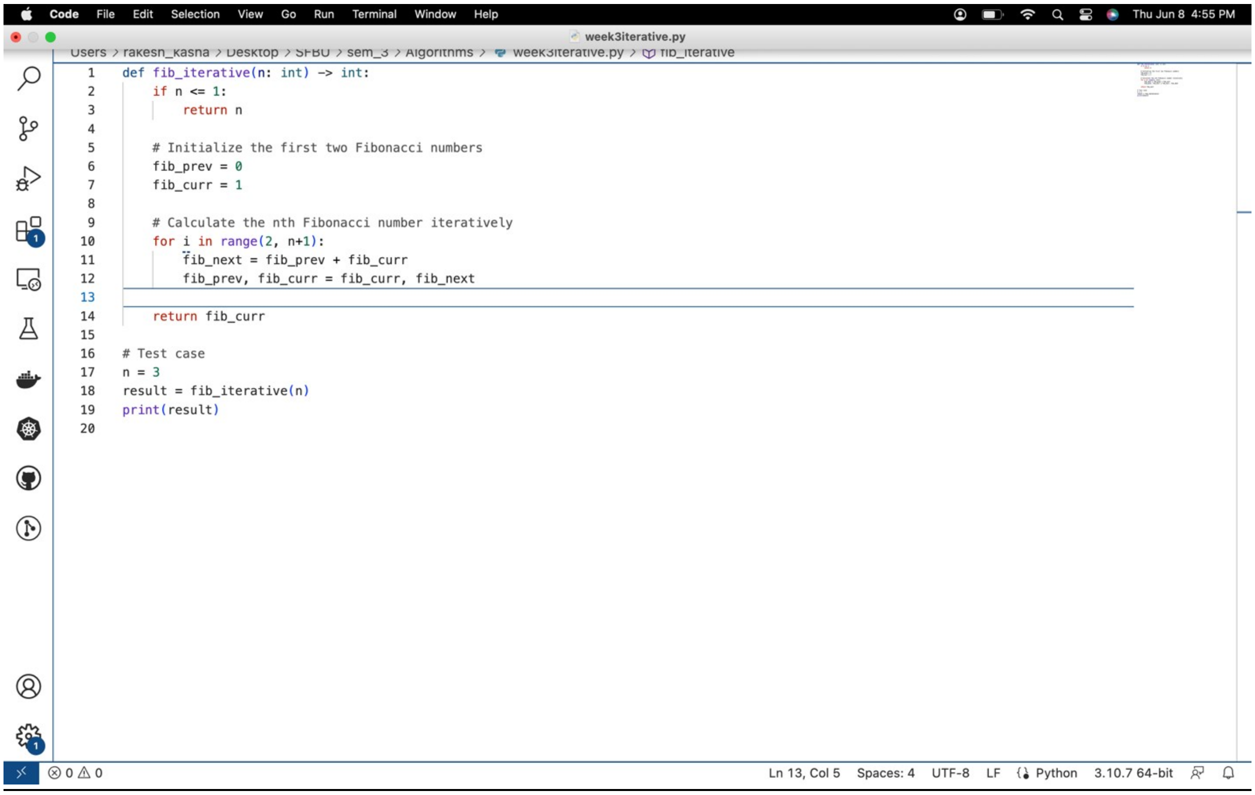
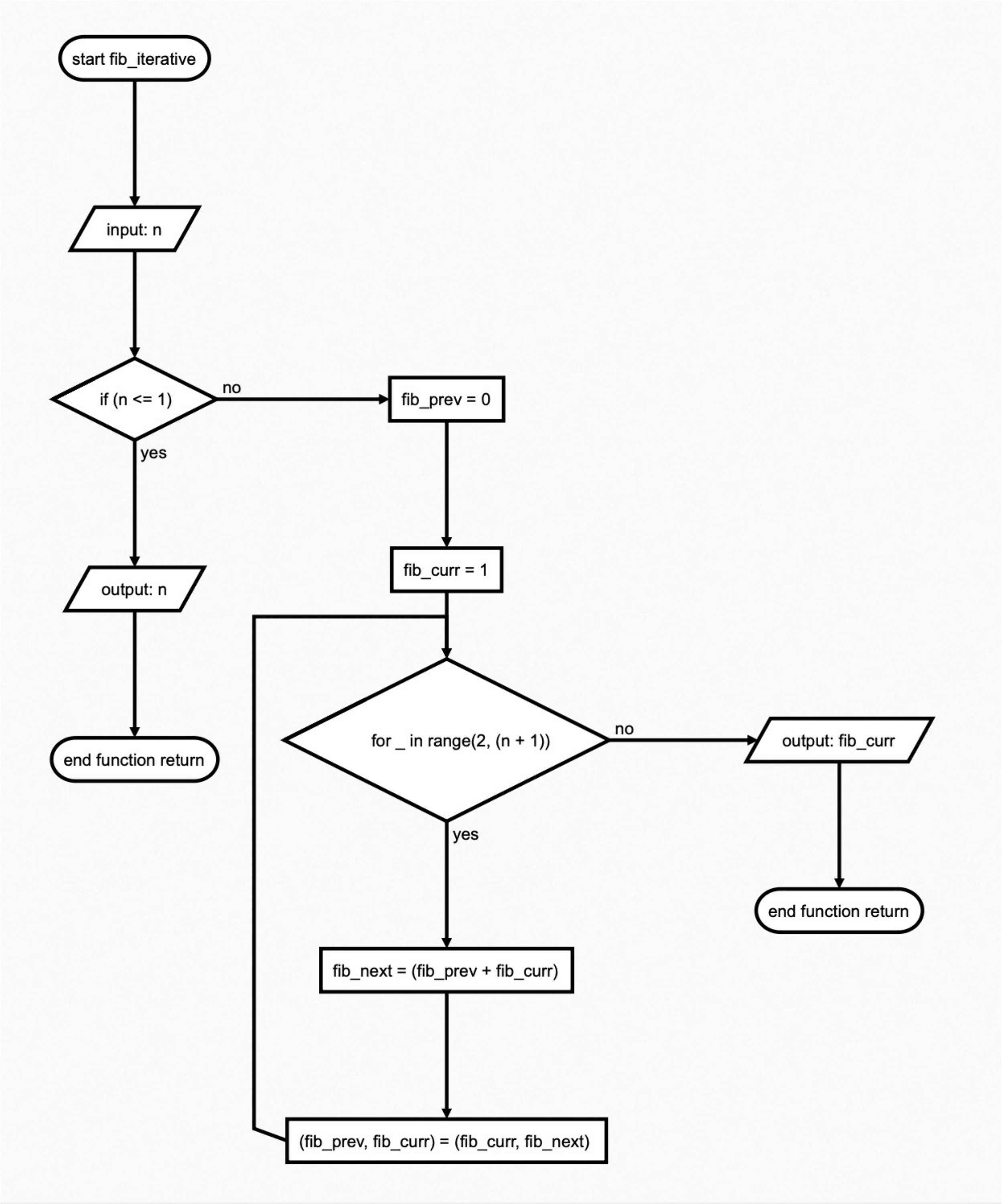
Homework3_q1

Itera0ve:

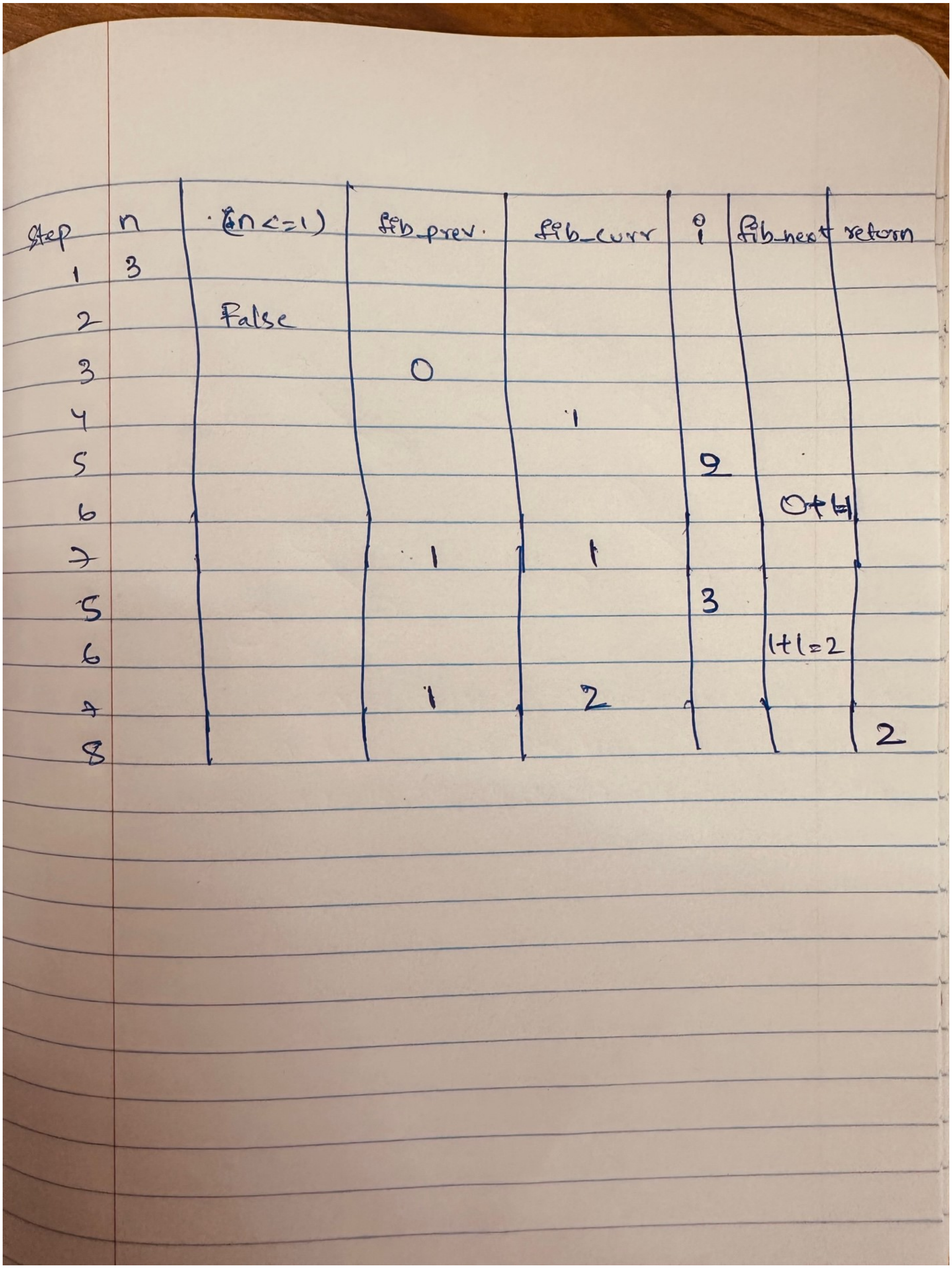


```python
def fib_iterative(n: int) -> int:
    if n <= 1:
        return n

    # Initialize the first two Fibonacci numbers
    fib_prev = 0
    fib_curr = 1

    # Calculate the nth Fibonacci number iteratively
    for i in range(2, n+1):
        fib_next = fib_prev + fib_curr
        fib_prev, fib_curr = fib_curr, fib_next

    return fib_curr

# Test case
n = 3
result = fib_iterative(n)
print(result)
```

Flowchart:

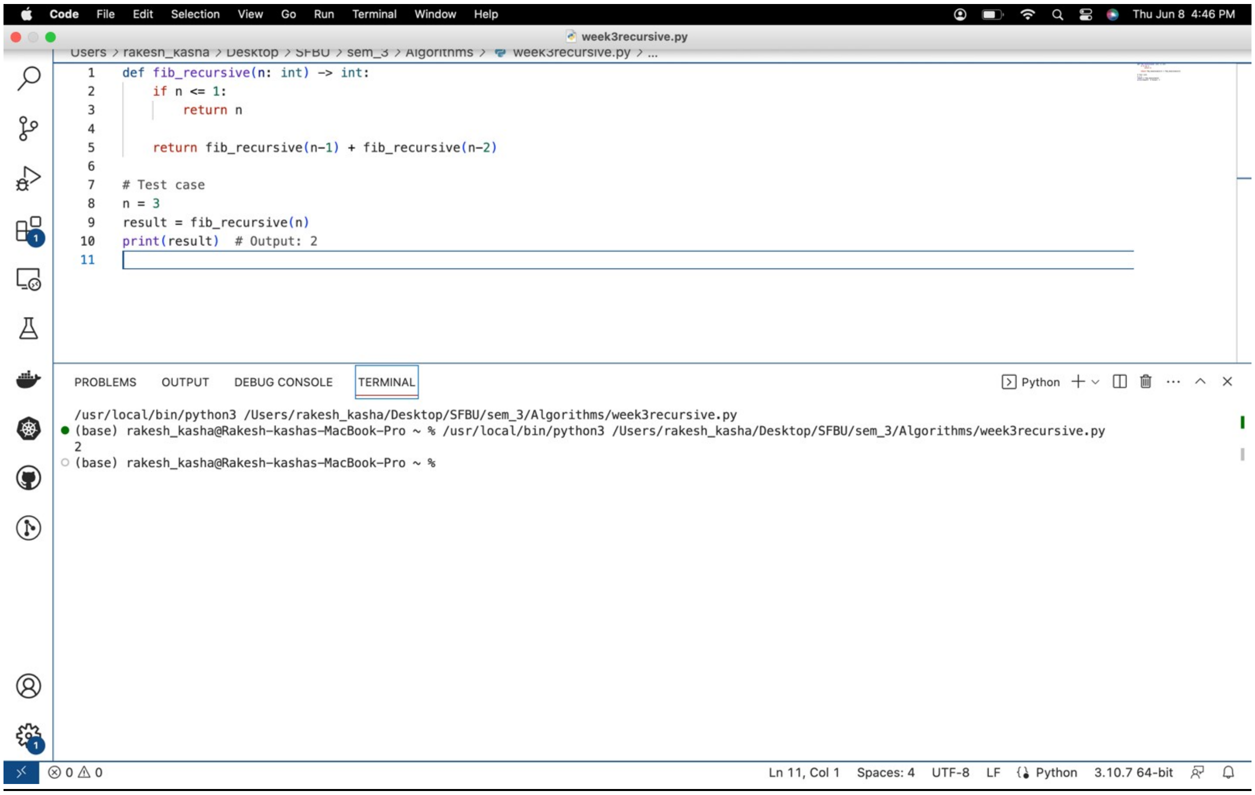

start fib_iterative
input: n
if (n <= 1)
no
yes
output: n
end function return
fib_prev = 0
fib_curr = 1
for _ in range(2, (n + 1))
no
yes
output: fib_curr
end function return
fib_next = (fib_prev + fib_curr)
(fib_prev, fib_curr) = (fib_curr, fib_next)

Trace table:



Step	n	(n<=1)	fib_prev.	fib_curr	i	fib_next	return
1	3						
2		False					
3			0				
4				1			
5					2		
6						0+1	
7			1	1			
5					3		
6						1+1=2	
7			1	2			
8							2

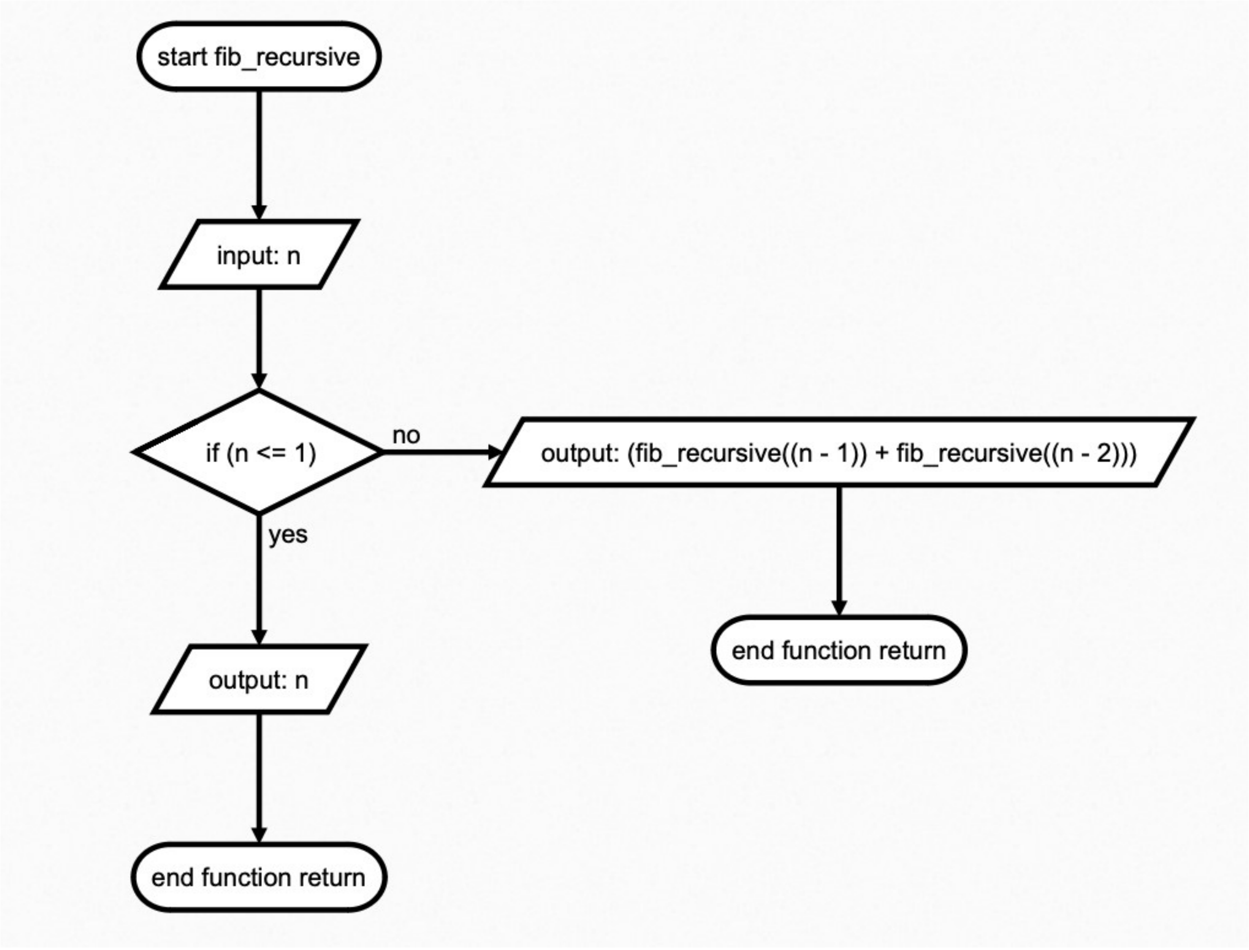
Recursive:



```python
def fib_recursive(n: int) -> int:
    if n <= 1:
        return n

    return fib_recursive(n-1) + fib_recursive(n-2)

# Test case
n = 3
result = fib_recursive(n)
print(result)  # Output: 2

```

```
/usr/local/bin/python3 /Users/rakesh_kasha/Desktop/SFBU/sem_3/Algorithms/week3recursive.py
● (base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ % /usr/local/bin/python3 /Users/rakesh_kasha/Desktop/SFBU/sem_3/Algorithms/week3recursive.py
2
○ (base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ %
```

Flowchart:


start fib_recursive
input: n
if (n <= 1)
no
output: (fib_recursive((n - 1)) + fib_recursive((n - 2)))
end function return
yes
output: n
end function return

Trace table:

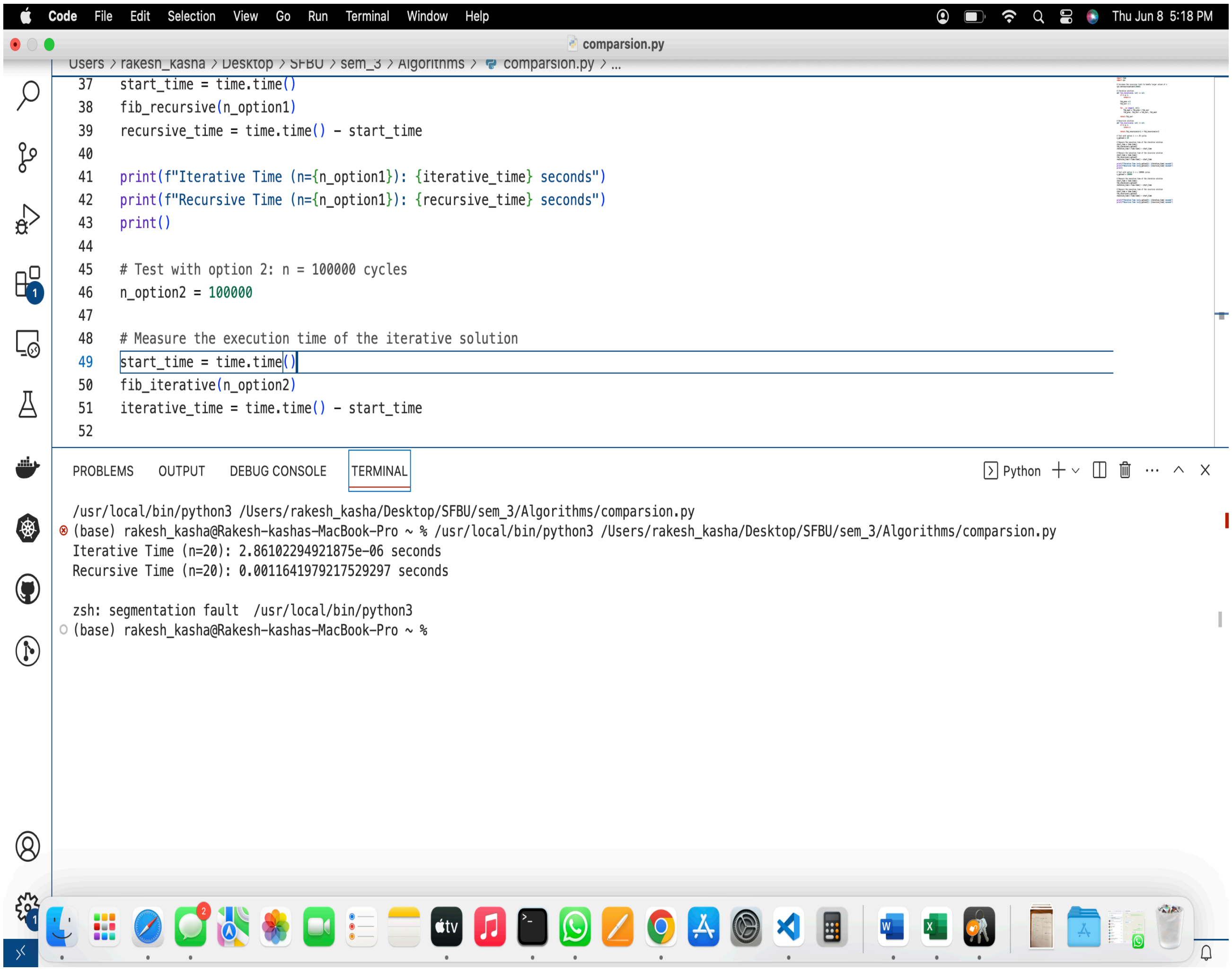
Recursive

step	n	n<=1	~~fib_recursive~~ return
1	3		
2		false	
4			fib_recursive(n-1) + fib_recur(n-2)
			(3-1) (3-2)
1	2		
2		False	
4			fib_Recursive (n-1) + fib_recursiv(n-2)
			(2-1) + (2-2)
1	1		
2		True	
3			1
4			1 + fib_recursic(0)
1	0		
2		True	
3			1
4			1+1 =2

Comparison code:

```
import time
import sys

# Increase the recursion limit to handle larger
values of n
sys.setrecursionlimit(10**5)

# Iterative solution
def fib_iterative(n: int) -> int:
    if n <= 1:
        return n

    fib_prev = 0
    fib_curr = 1

    for _ in range(2, n+1):
        fib_next = fib_prev + fib_curr
        fib_prev, fib_curr = fib_curr, fib_next

    return fib_curr

# Recursive solution
def fib_recursive(n: int) -> int:
    if n <= 1:
        return n

    return fib_recursive(n-1) + fib_recursive(n-2)

# Test with option 1: n = 20 cycles
n_option1 = 20

# Measure the execution time of the iterative
solution
start_time = time.time()
fib_iterative(n_option1)
iterative_time = time.time() - start_time

# Measure the execution time of the recursive
solution
start_time = time.time()
fib_recursive(n_option1)
recursive_time = time.time() - start_time

print(f"Iterative Time (n={n_option1}):
{iterative_time} seconds")
```

```python
print(f"Recursive Time (n={n_option1}): {recursive_time} seconds")
print()

# Test with option 2: n = 100000 cycles
n_option2 = 100000

# Measure the execution time of the iterative solution
start_time = time.time()
fib_iterative(n_option2)
iterative_time = time.time() - start_time

# Measure the execution time of the recursive solution
start_time = time.time()
fib_recursive(n_option2)
recursive_time = time.time() - start_time

print(f"Iterative Time (n={n_option2}): {iterative_time} seconds")
print(f"Recursive Time (n={n_option2}): {recursive_time} seconds")
```



```python
start_time = time.time()
fib_recursive(n_option1)
recursive_time = time.time() - start_time

print(f"Iterative Time (n={n_option1}): {iterative_time} seconds")
print(f"Recursive Time (n={n_option1}): {recursive_time} seconds")
print()

# Test with option 2: n = 100000 cycles
n_option2 = 100000

# Measure the execution time of the iterative solution
start_time = time.time()
fib_iterative(n_option2)
iterative_time = time.time() - start_time

```

```
/usr/local/bin/python3 /Users/rakesh_kasha/Desktop/SFBU/sem_3/Algorithms/comparsion.py
(base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ % /usr/local/bin/python3 /Users/rakesh_kasha/Desktop/SFBU/sem_3/Algorithms/comparsion.py
Iterative Time (n=20): 2.86102294921875e-06 seconds
Recursive Time (n=20): 0.0011641979217529297 seconds

zsh: segmentation fault  /usr/local/bin/python3
(base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ %
```

Table Results:

Option	**Iterative**	**Recursive**
n = 20 cycles	The program's execution time is 2.861	The program's execution time is 0.001
n = 100000 cycles	The program got crashed	The program got crashed
Big-O	O(n)	O(2^n)