Lab Assignment – 01

Name:G.Harika

HT.NO:2303A51612

Batch:05

Task-01:

Prompt:

Find Fibanocci series upto n terms without using user defined functions

Code:

```
n = int(input("Enter the number of terms: "))

a, b = 0, 1

count = 0

if n <= 0:

   print("Please enter a positive integer")

elif n == 1:

   print("Fibonacci sequence upto", n, "term:")

   print(a)

else:

   print("Fibonacci sequence upto", n, "terms:")

   while count < n:

      print(a, end=' ')
```

```
    c = a + b
    a = b
    b = c
    count += 1
```

Output:

```
Enter the number of terms: 5
Fibonacci sequence upto 5 terms:
0 1 1 2 3
```

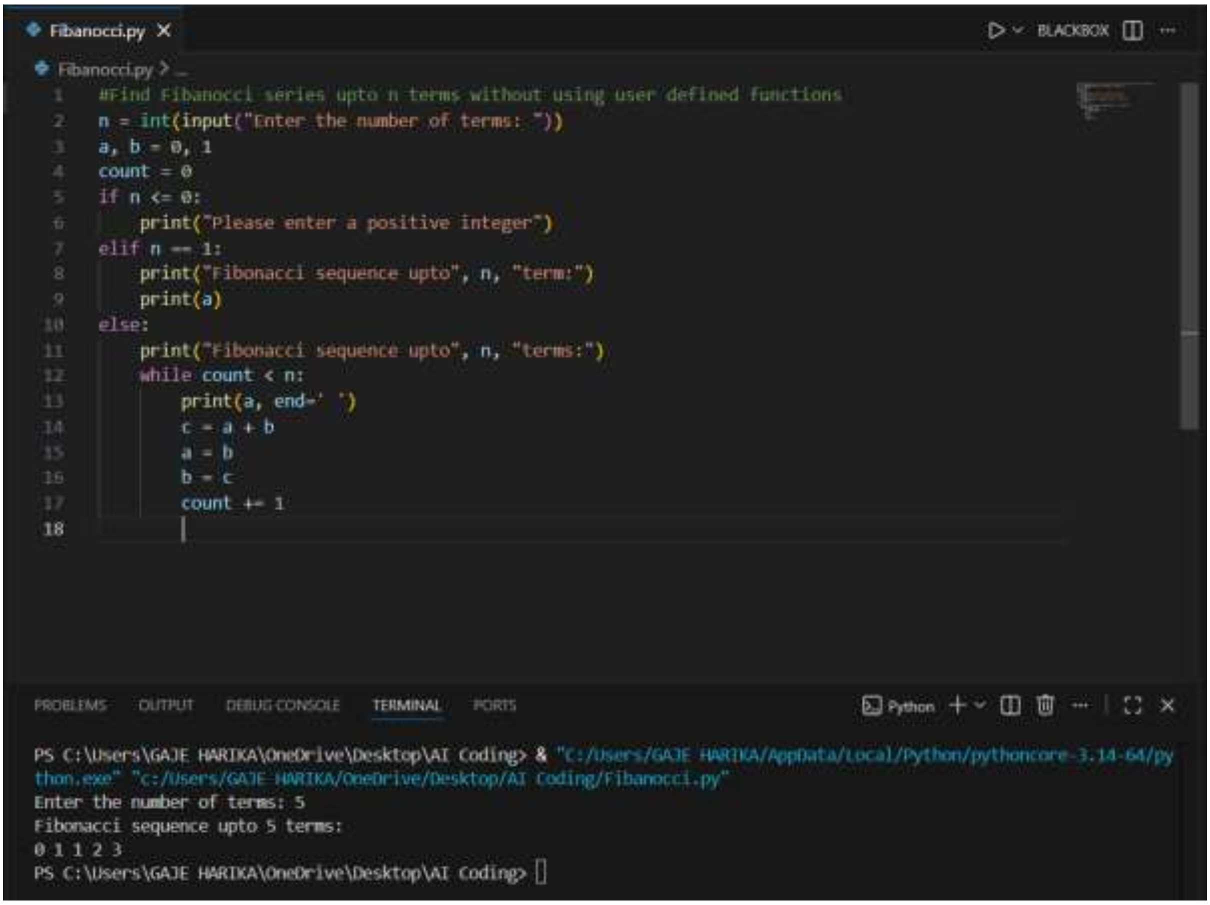
Implementation:

1. The program asks for n and starts the Fibonacci series with a = 0 and b = 1.

2. If n is 0 or negative, it shows an error; if n == 1, it prints only 0.

3. Otherwise, it runs a while loop, printing a each time and updating the values using c = a + b then a = b, b =c and count increases by 1.

4. The loop stops after printing n numbers.



```python
#Find Fibanocci series upto n terms without using user defined functions
n = int(input("Enter the number of terms: "))
a, b = 0, 1
count = 0
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(a)
else:
    print("Fibonacci sequence upto", n, "terms:")
    while count < n:
        print(a, end=' ')
        c = a + b
        a = b
        b = c
        count += 1

```

```
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> & "C:/Users/GAJE HARIKA/AppData/Local/Python/pythoncore-3.14-64/python.exe" "c:/Users/GAJE HARIKA/OneDrive/Desktop/AI Coding/Fibanocci.py"
Enter the number of terms: 5
Fibonacci sequence upto 5 terms:
0 1 1 2 3
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding>
```

Task-02:

Prompt:

Find Fibanocci series upto n terms optimized code without using functions

Code:

```python
n = int(input("Enter the number of terms: "))

a, b = 0, 1

if n <= 0:

   print("Please enter a positive integer")

elif n == 1:

   print("Fibonacci sequence upto", n, "term:")
```

```
    print(a)
else:
    print("Fibonacci sequence upto", n, "terms:")
    for _ in range(n):
        print(a, end=' ')
        a, b = b, a + b
```

Output:

```
Enter the number of terms: 7
Fibonacci sequence upto 7 terms:
0 1 1 2 3 5 8
```

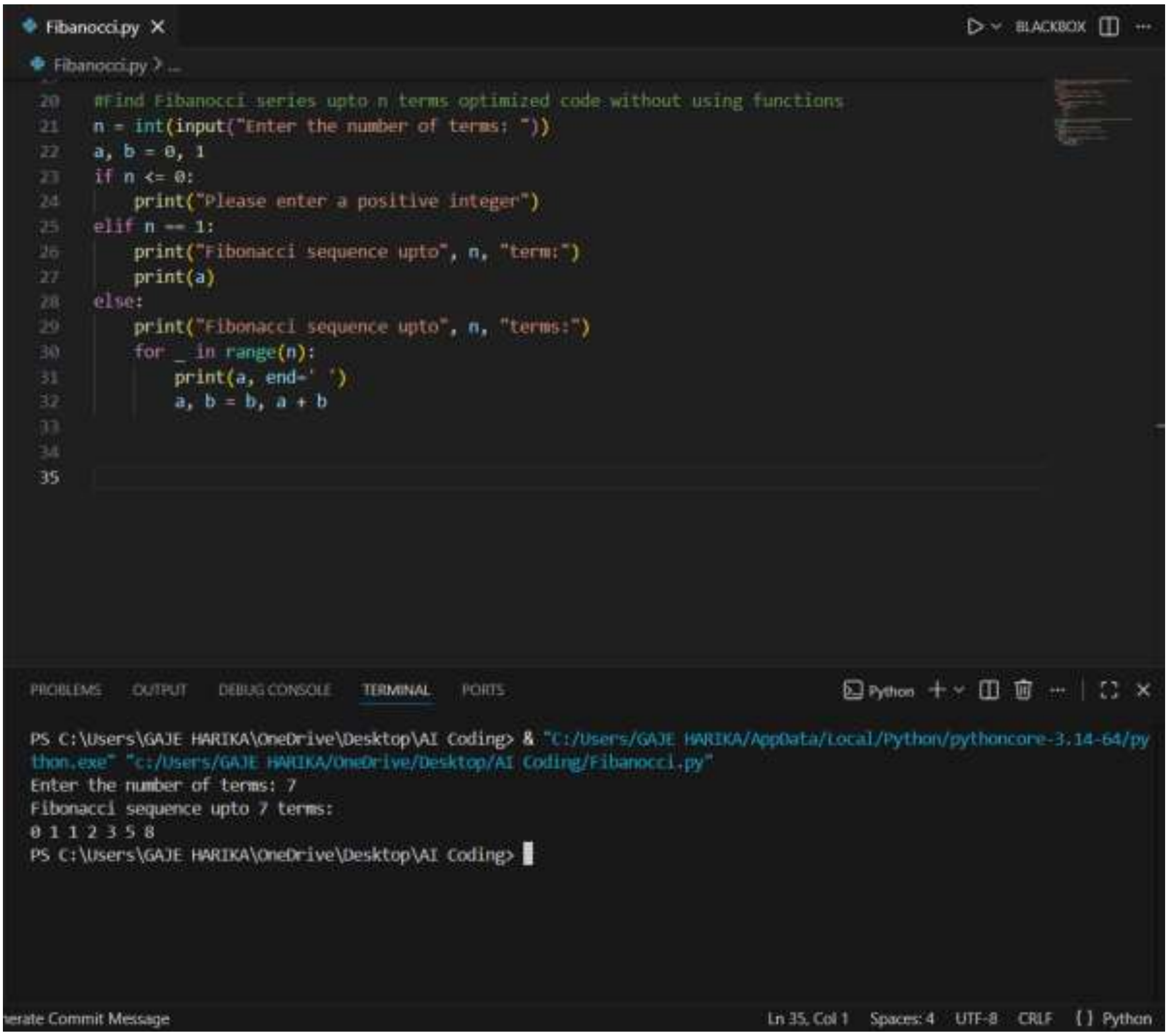
Implementation:

1. The program takes n from the user and starts with a = 0 and b = 1.

2. If n is not positive, it shows an error message.

3. Otherwise, a for loop runs n times, printing a each time.

4. Inside the loop, the next Fibonacci number is generated using a, b = b, a + b.



```python
#Find Fibanocci series upto n terms optimized code without using functions
n = int(input("Enter the number of terms: "))
a, b = 0, 1
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(a)
else:
    print("Fibonacci sequence upto", n, "terms:")
    for _ in range(n):
        print(a, end=' ')
        a, b = b, a + b
```

```
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> & "C:/Users/GAJE HARIKA/AppData/Local/Python/pythoncore-3.14-64/py
thon.exe" "c:/Users/GAJE HARIKA/OneDrive/Desktop/AI Coding/Fibanocci.py"
Enter the number of terms: 7
Fibonacci sequence upto 7 terms:
0 1 1 2 3 5 8
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> 
```

Task-03:

Prompt:

Find Fibanocci series using Functions returns or prints the sequence upto n terms

Code:

```python
def fibonacci_series(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
n = int(input("Enter the number of terms: "))
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(fibonacci_series(n))
else:
    print("Fibonacci sequence upto", n, "terms:")
    print(fibonacci_series(n))
```

Output:

```
Enter the number of terms: 7
Fibonacci sequence upto 7 terms:
[0, 1, 1, 2, 3, 5, 8]
```

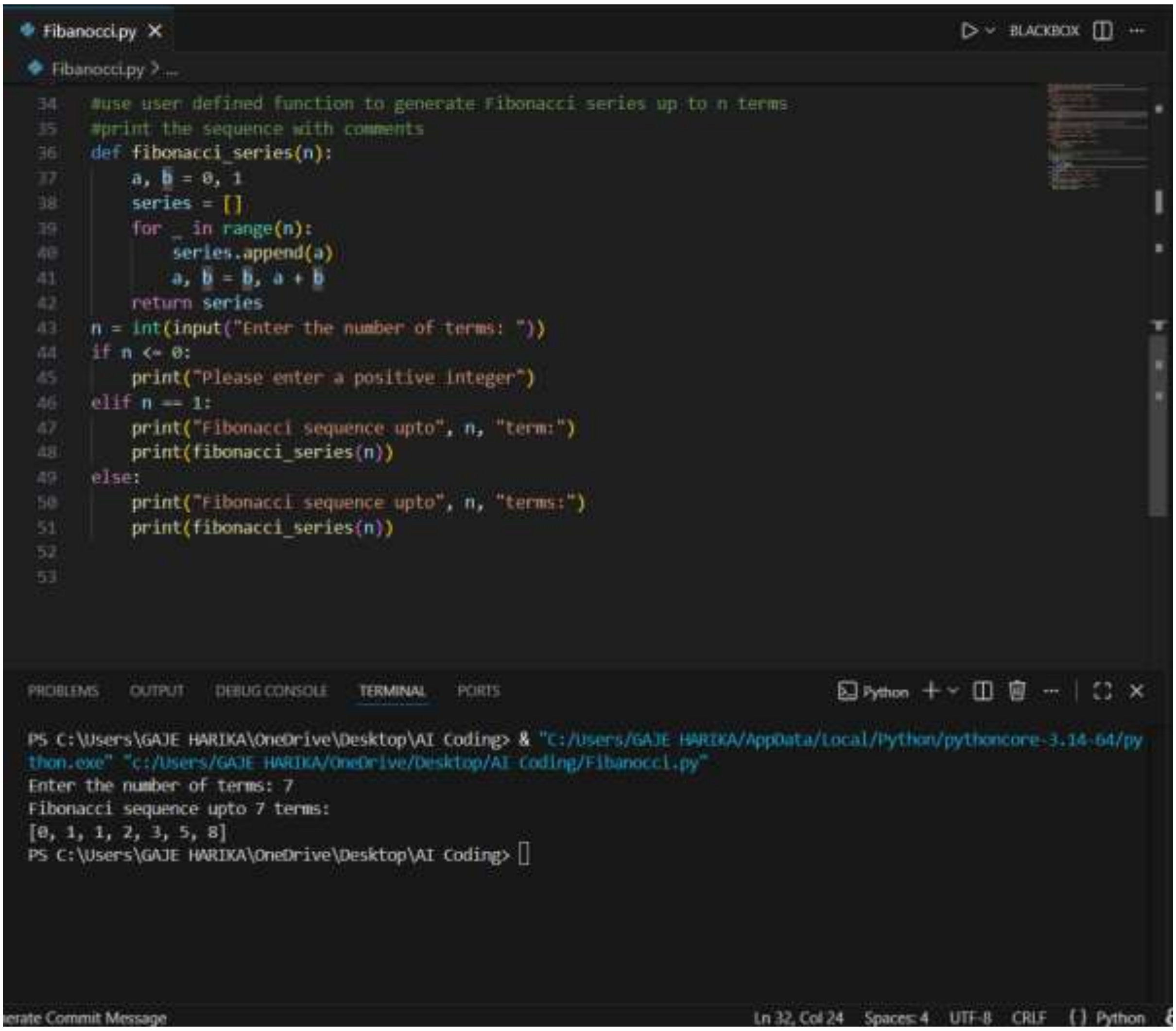
Implementation:

1. The function starts with 0 and 1 as the first Fibonacci numbers.

2. It repeats n times to create the next numbers.

3. Each number is added to a list.

4. The list is returned and printed as the Fibonacci sequence.



```python
#use user defined function to generate Fibonacci series up to n terms
#print the sequence with comments
def fibonacci_series(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
n = int(input("Enter the number of terms: "))
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(fibonacci_series(n))
else:
    print("Fibonacci sequence upto", n, "terms:")
    print(fibonacci_series(n))
```

```
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> & "C:/Users/GAJE HARIKA/AppData/Local/Python/pythoncore-3.14-64/py
thon.exe" "c:/Users/GAJE HARIKA/OneDrive/Desktop/AI Coding/Fibanocci.py"
Enter the number of terms: 7
Fibonacci sequence upto 7 terms:
[0, 1, 1, 2, 3, 5, 8]
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> 
```

<u>Task-04:</u>

<u>Prompt:</u>

# Fibonacci series with Procedural vs Modular Fibonacci Code AI code with and without

# Procedural approach

<u>Code:</u>

```python
def fibonacci_series(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
n = int(input("Enter the number of terms: "))
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(fibonacci_series(n))
else:
    print("Fibonacci sequence upto", n, "terms:")
    print(fibonacci_series(n))
```

Output:

```
Enter the number of terms: 5
Fibonacci sequence upto 5 terms:
[0, 1, 1, 2, 3]
```

Prompt:

```
# Modular approach
```

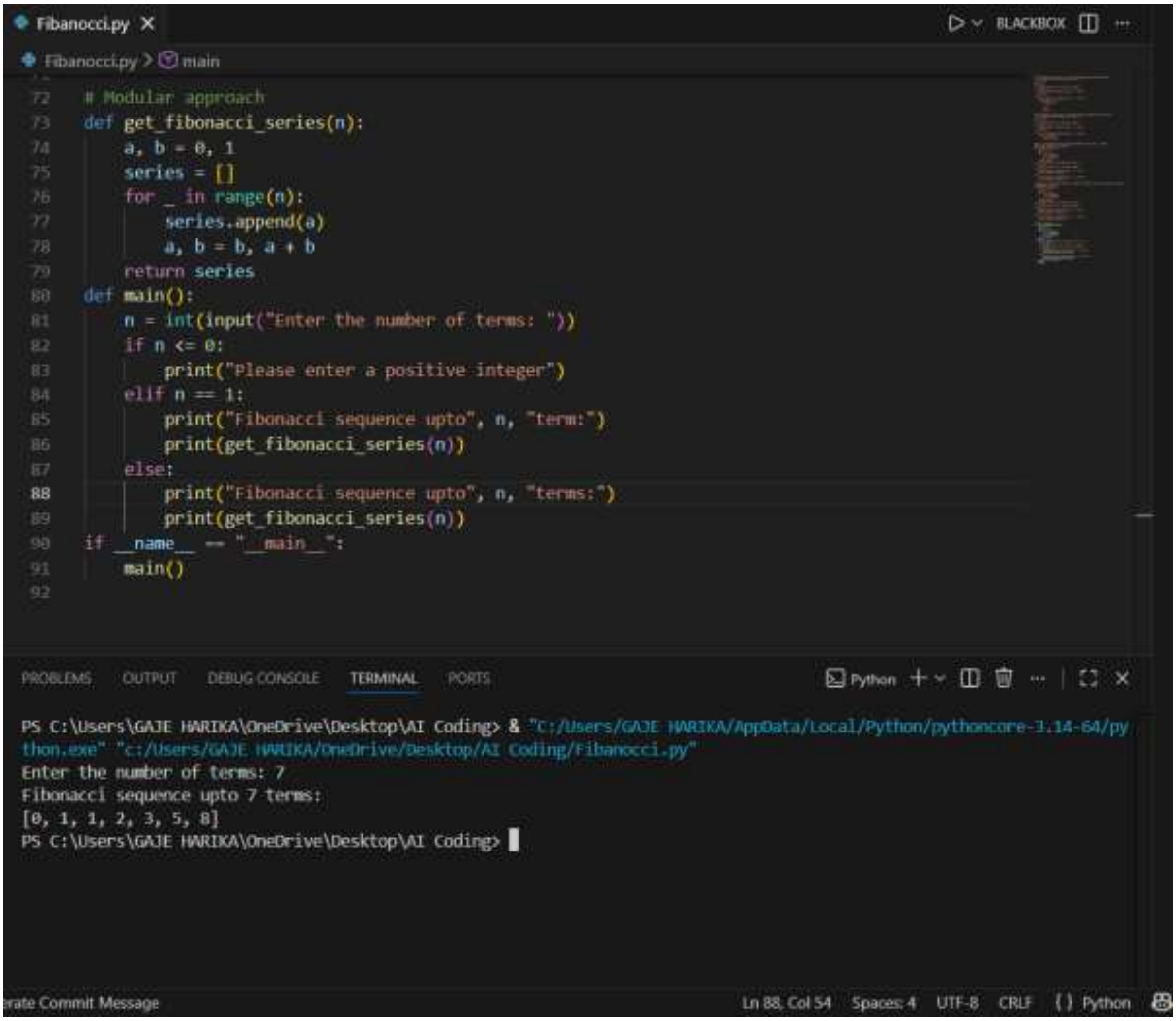
Code:

```
def get_fibonacci_series(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
def main():
    n = int(input("Enter the number of terms: "))
    if n <= 0:
        print("Please enter a positive integer")
    elif n == 1:
        print("Fibonacci sequence upto", n, "term:")
        print(get_fibonacci_series(n))
    else:
        print("Fibonacci sequence upto", n, "terms:")
        print(get_fibonacci_series(n))
if __name__ == "__main__":
    main()
```

Output:

```
Enter the number of terms: 7
Fibonacci sequence upto 7 terms:
[0, 1, 1, 2, 3, 5, 8]
```

Implementation:



```python
# Modular approach
def get_fibonacci_series(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
def main():
    n = int(input("Enter the number of terms: "))
    if n <= 0:
        print("Please enter a positive integer")
    elif n == 1:
        print("Fibonacci sequence upto", n, "term:")
        print(get_fibonacci_series(n))
    else:
        print("Fibonacci sequence upto", n, "terms:")
        print(get_fibonacci_series(n))
if __name__ == "__main__":
    main()
```

```
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> & "C:/Users/GAJE HARIKA/AppData/Local/Python/pythoncore-3.14-64/py
thon.exe" "c:/Users/GAJE HARIKA/OneDrive/Desktop/AI Coding/Fibanocci.py"
Enter the number of terms: 7
Fibonacci sequence upto 7 terms:
[0, 1, 1, 2, 3, 5, 8]
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding>
```

Task-05:

Prompt:

#AI-Generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches for Fibonacci Series)

# Iterative approach

Code:

```
def fibonacci_iterative(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
n = int(input("Enter the number of terms: "))
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(fibonacci_iterative(n))
else:
    print("Fibonacci sequence upto", n, "terms:")
    print(fibonacci_iterative(n))
```

Code:

```
Enter the number of terms: 8
Fibonacci sequence upto 8 terms:
[0, 1, 1, 2, 3, 5, 8, 13]
```

Implementation:

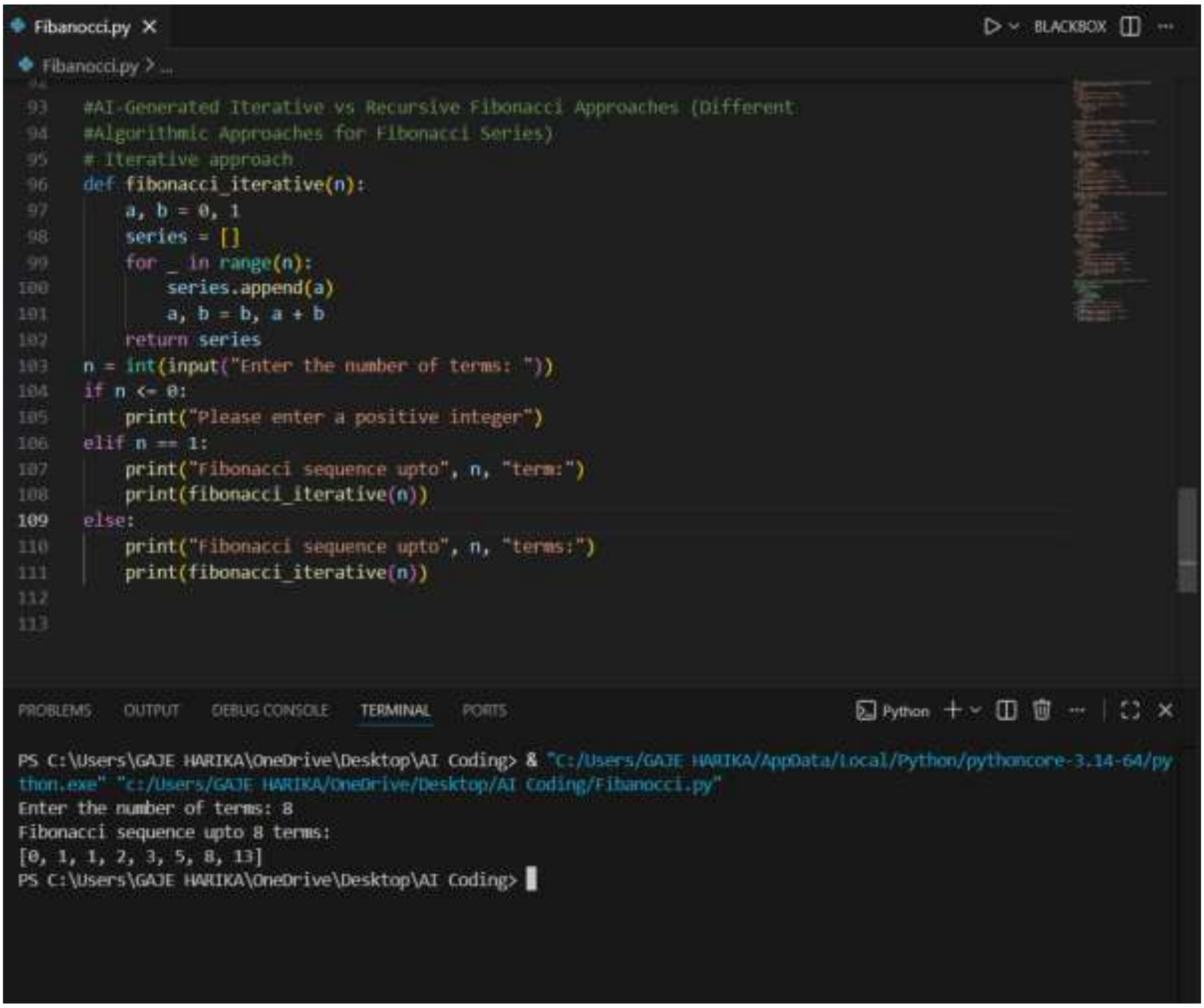
1.The function is defined to generate Fibonacci numbers.

2. a and b are initialized to 0 and 1.

3. A loop runs n times.

4. Each Fibonacci number is added to a list.

5.The list is returned and printed.



```
#AI-Generated Iterative vs Recursive Fibonacci Approaches (Different
#Algorithmic Approaches for Fibonacci Series)
# Iterative approach
def fibonacci_iterative(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
n = int(input("Enter the number of terms: "))
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(fibonacci_iterative(n))
else:
    print("Fibonacci sequence upto", n, "terms:")
    print(fibonacci_iterative(n))
```

```
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> & "C:/Users/GAJE HARIKA/AppData/Local/Python/pythoncore-3.14-64/py
thon.exe" "c:/Users/GAJE HARIKA/OneDrive/Desktop/AI Coding/Fibanocci.py"
Enter the number of terms: 8
Fibonacci sequence upto 8 terms:
[0, 1, 1, 2, 3, 5, 8, 13]
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding>
```

Prompt:

# Recursive approach

Code:

```
def fibonacci_recursive(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    else:
        series = fibonacci_recursive(n - 1)
        series.append(series[-1] + series[-2])
        return series
n = int(input("Enter the number of terms: "))
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(fibonacci_recursive(n))
else:
    print("Fibonacci sequence upto", n, "terms:")
    print(fibonacci_recursive(n))
```

Output:

```
Enter the number of terms: 6
Fibonacci sequence upto 6 terms:
[0, 1, 1, 2, 3, 5]
```

Implementation:

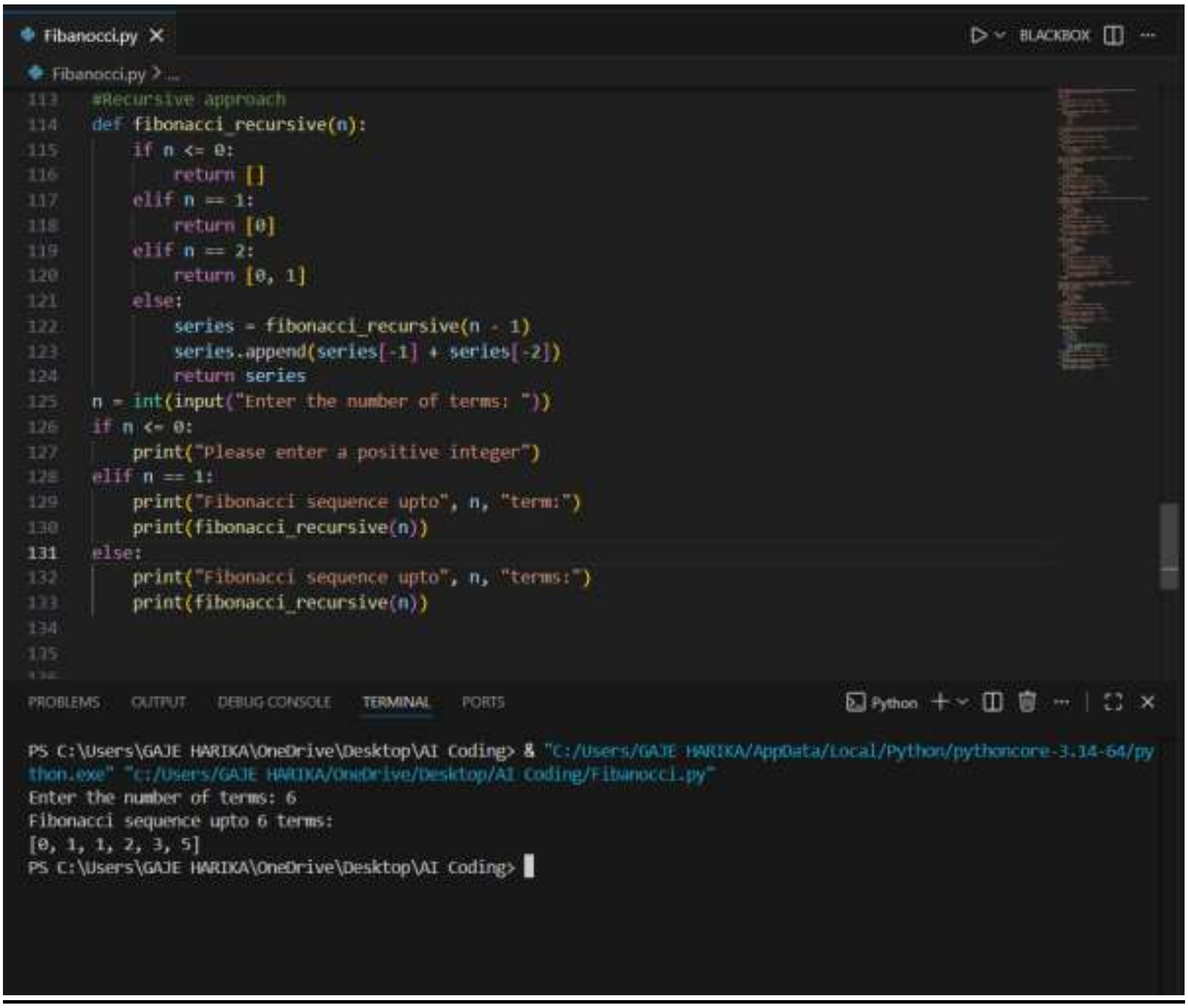
1. The function fibonacci_recursive(n) is called.

2. If n is 1 or 2, it returns the base Fibonacci values.

3. If n is greater than 2, the function calls itself with n-1.

4. The last two numbers are added to get the next Fibonacci number.

5. The updated list is returned and printed.



```python
#Recursive approach
def fibonacci_recursive(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    else:
        series = fibonacci_recursive(n - 1)
        series.append(series[-1] + series[-2])
        return series
n = int(input("Enter the number of terms: "))
if n <= 0:
    print("Please enter a positive integer")
elif n == 1:
    print("Fibonacci sequence upto", n, "term:")
    print(fibonacci_recursive(n))
else:
    print("Fibonacci sequence upto", n, "terms:")
    print(fibonacci_recursive(n))
```

```
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> & "C:/Users/GAJE HARIKA/AppData/Local/Python/pythoncore-3.14-64/python.exe" "c:/Users/GAJE HARIKA/OneDrive/Desktop/AI Coding/Fibanocci.py"
Enter the number of terms: 6
Fibonacci sequence upto 6 terms:
[0, 1, 1, 2, 3, 5]
PS C:\Users\GAJE HARIKA\OneDrive\Desktop\AI Coding> 
```