

Lab Assignment-01

Prompt:

#find fibonacci 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, ":")
    print(a)
else:
    print("Fibonacci sequence:")
    while count < n:
        print(a, end=' ')
        a, b = b, a + b
        count += 1
```

Output:

Enter the number of terms: 5

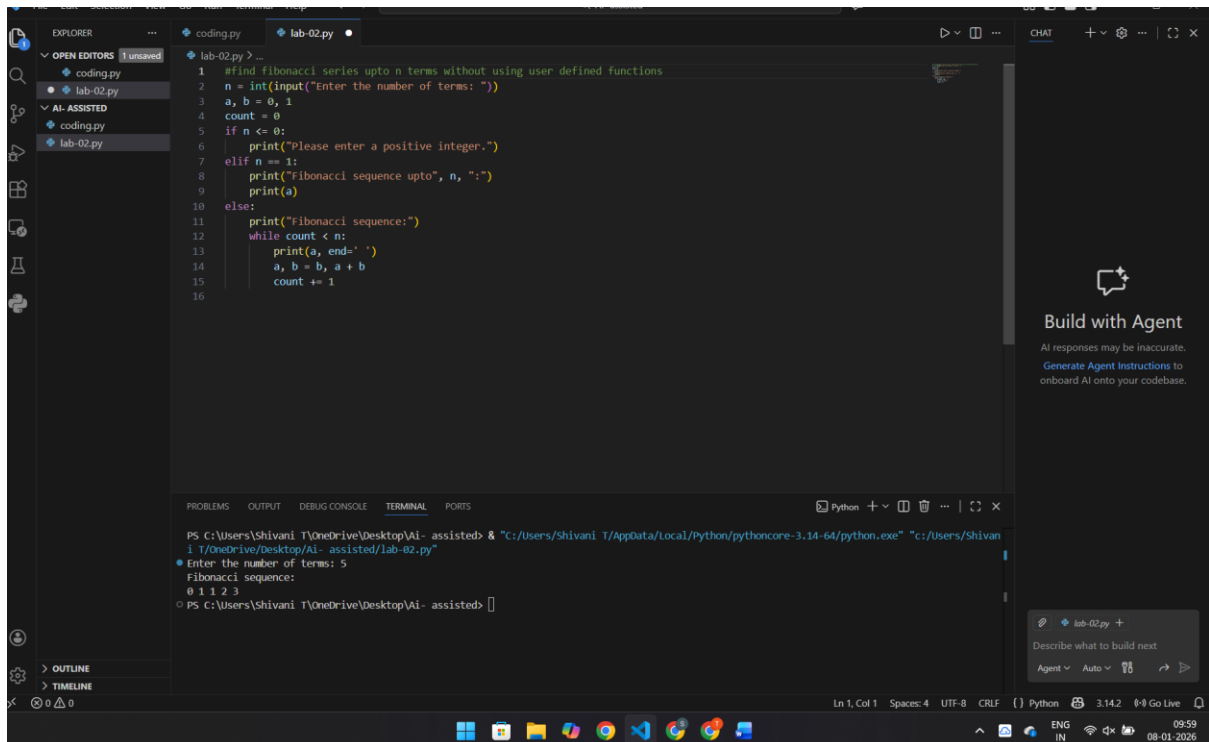
Fibonacci sequence:

0 1 1 2 3

Explanation:

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 $a, b = b, a + b$.
4. The loop stops after printing n numbers.



```
1 #find fibonacci series upto n terms without using user defined functions
2 n = int(input("Enter the number of terms: "))
3 a, b = 0, 1
4 count = 0
5 if n <= 0:
6     print("Please enter a positive integer.")
7 elif n == 1:
8     print("Fibonacci sequence upto", n, ":")
9     print(a)
10 else:
11     print("Fibonacci sequence:")
12     while count < n:
13         print(a, end=' ')
14         a, b = b, a + b
15         count += 1
16
```

PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted> & "C:\Users\Shivani T\AppData\Local\Python\pythoncore-3.14-64\python.exe" "C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted\lab-02.py"

Enter the number of terms: 5

Fibonacci sequence:

0 1 1 2 3

PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted>

Prompt:

#find fibonacci series upto n terms without using user defined functions

Code:

```
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, ":")
```

```
    print(a)
```

else:

```
print("Fibonacci sequence:")
```

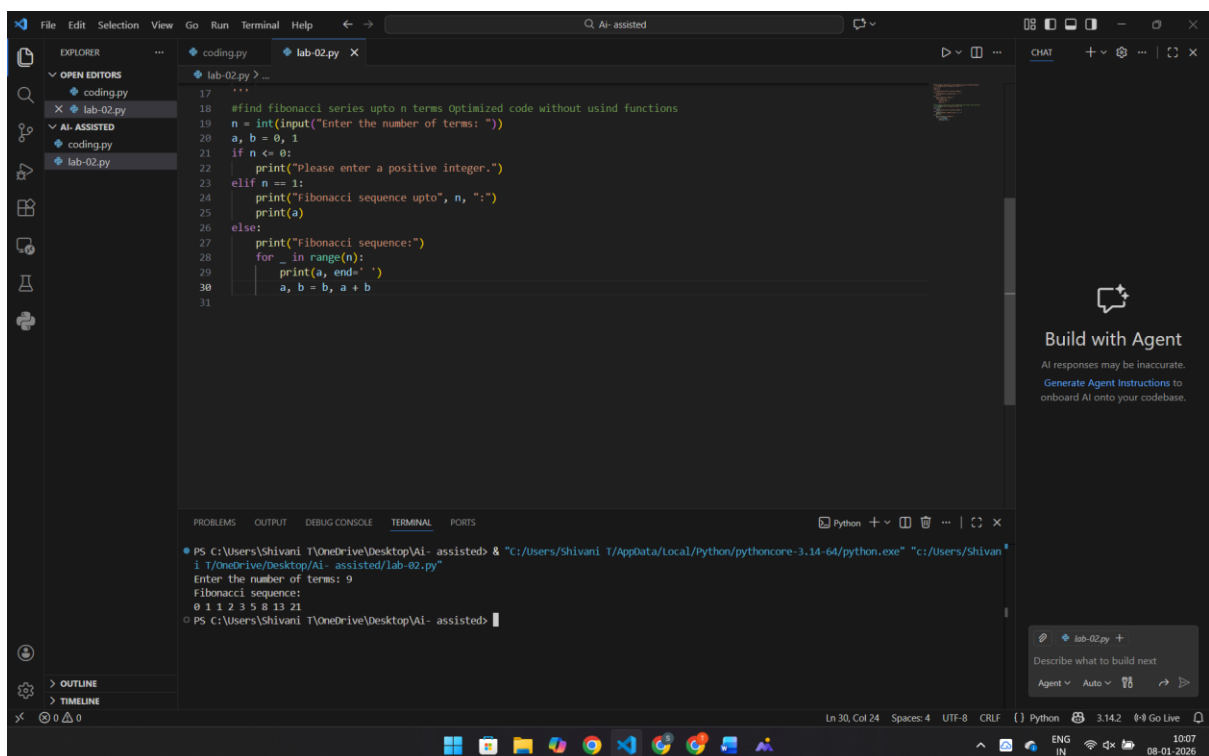
```
for _ in range(n):
```

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

```
    a, b = b, a + b
```

Explanation:

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.



```
17 '''
18 #find fibonacci series upto n terms Optimized code without using functions
19 n = int(input("Enter the number of terms: "))
20 a, b = 0, 1
21 if n <= 0:
22     print("Please enter a positive integer.")
23 elif n == 1:
24     print("Fibonacci sequence upto", n, ":")
25     print(a)
26 else:
27     print("Fibonacci sequence:")
28     for _ in range(n):
29         print(a, end=' ')
30         a, b = b, a + b
31
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Shivani T\OneDrive\Desktop\AI- assisted> & "c:/Users/Shivani T/AppData/Local/Python/pythoncore-3.14-64/python.exe" "c:/Users/Shivani T/OneDrive/Desktop/AI- assisted/lab-02.py"
Enter the number of terms: 9
Fibonacci sequence:
0 1 1 2 3 5 8 13 21
PS C:\Users\Shivani T\OneDrive\Desktop\AI- assisted>
```

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

lab-02.py +

Describe what to build next

Agent Auto

Ln 30, Col 24 Spaces: 4 UTF-8 CRLF Python 3.14.2 Go Live

1007 08-01-2026

Prompt:

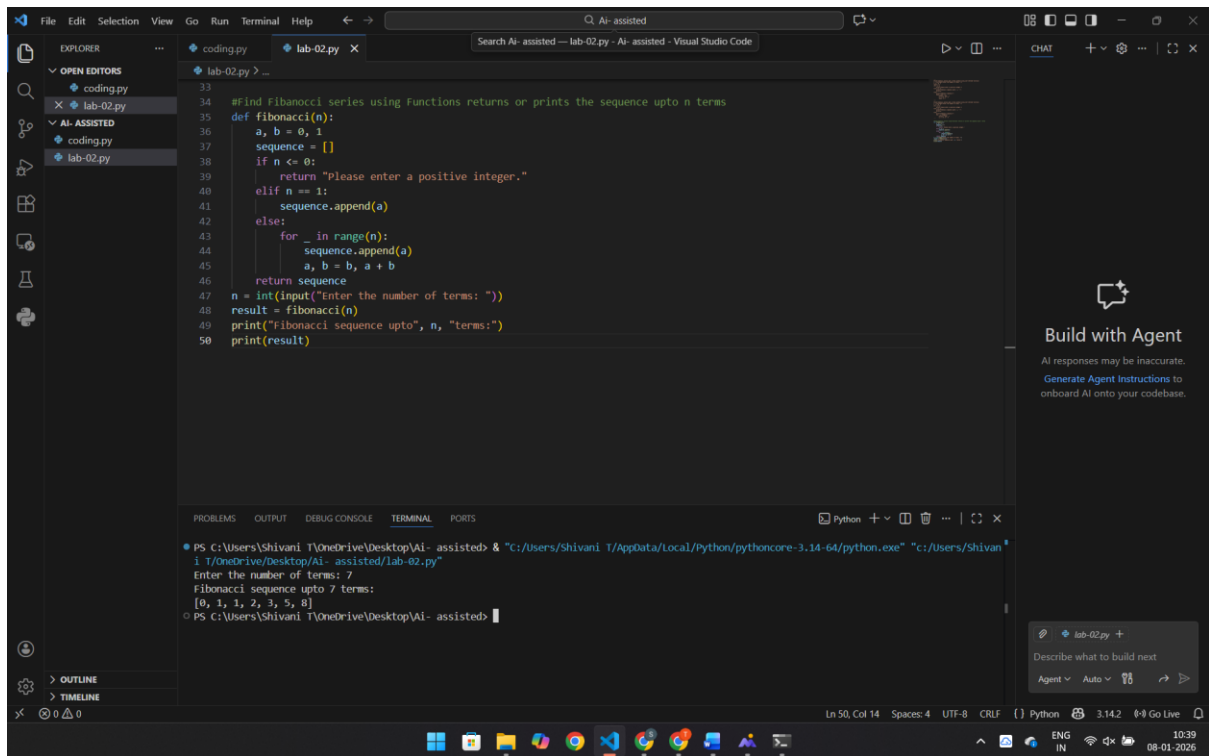
Find Fibanocci series using Functions returns or prints the sequence upto n terms

Code:

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

Explanation:

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.



Prompt:

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

Procedural approach

Code:

```
n = int(input("Enter the number of terms in modular fibonacci series: "))
```

```
a, b = 0, 1
```

```
if n <= 0:
```

```
    print("Please enter a positive integer.")
```

```
else:
```

```
    print("Fibonacci sequence up to", n, "terms:")
```

```
    for _ in range(n):
```

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

```
        a, b = b, a + b
```

```

print()

# Modular approach

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 in modular fibonacci series: "))
if n <= 0:
    print("Please enter a positive integer.")
else:
    print("Fibonacci sequence up to", n, "terms:")
    print(fibonacci_series(n))

```

output:

Enter the number of terms in modular fibonacci series: 5

Fibonacci sequence up to 5 terms:

0 1 1 2 3

Explanation:

1.First part prints Fibonacci numbers directly using a for loop: it starts with a = 0, b = 1,

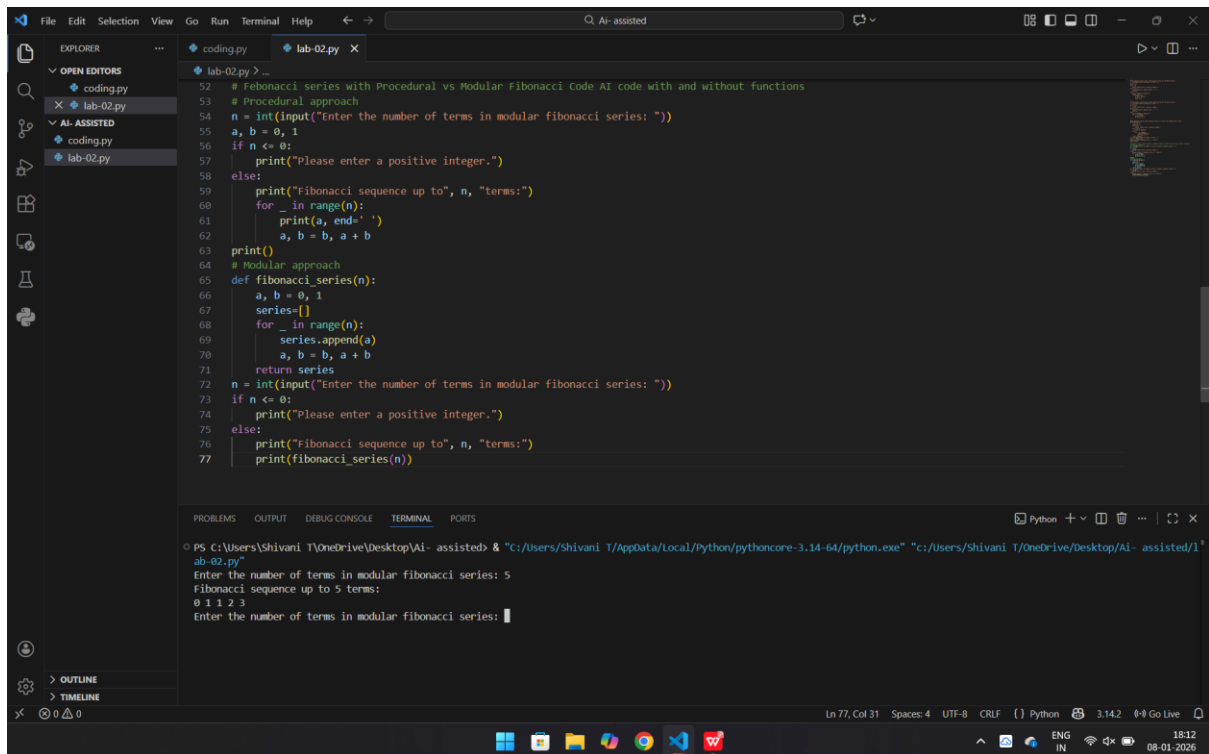
prints a each time, then updates with a, b = b, a + b.

2.It checks if n is positive before printing the series.

3.The second part uses a modular (function) approach: fibonacci_series(n) builds the

sequence in a list and returns it.

4.The program again asks for n, validates it, calls the function, and prints the returned list



The screenshot shows a VS Code editor with a Python file named 'lab-02.py'. The code implements two approaches to generate the Fibonacci series: a procedural approach and a modular approach using a function. The terminal output shows the program running, asking for the number of terms (5), and displaying the resulting Fibonacci sequence: 0 1 1 2 3.

```
52 # Fibonacci series with Procedural vs Modular Fibonacci code AI code with and without functions
53 # Procedural approach
54 n = int(input("Enter the number of terms in modular fibonacci series: "))
55 a, b = 0, 1
56 if n <= 0:
57     print("Please enter a positive integer.")
58 else:
59     print("Fibonacci sequence up to", n, "terms:")
60     for _ in range(n):
61         print(a, end=' ')
62         a, b = b, a + b
63     print()
64 # Modular approach
65 def fibonacci_series(n):
66     a, b = 0, 1
67     series = []
68     for _ in range(n):
69         series.append(a)
70         a, b = b, a + b
71     return series
72 n = int(input("Enter the number of terms in modular fibonacci series: "))
73 if n <= 0:
74     print("Please enter a positive integer.")
75 else:
76     print("Fibonacci sequence up to", n, "terms:")
77     print(fibonacci_series(n))
```

Terminal Output:

```
PS C:\Users\Shivani T\OneDrive\Desktop\AI- assisted> "C:\Users\Shivani T\AppData\Local\Python\pythoncore-3.14-64\python.exe" "C:\Users\Shivani T\OneDrive\Desktop\AI- assisted\lab-02.py"
Enter the number of terms in modular fibonacci series: 5
Fibonacci sequence up to 5 terms:
0 1 1 2 3
Enter the number of terms in modular fibonacci series: 
```

Prompt:

#AI-Generated Iterative vs Recursive Fibonacci Approaches (Different

#Algorithmic Approaches for Fibonacci Series)

Iterative approach and Recursive 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

def fibonacci_recursive(n, a=0, b=1, series=None):

    if series is None:

        series = []

    if n == 0:

        return series

    series.append(a)

    return fibonacci_recursive(n - 1, b, a + b, series)

n = int(input("Enter the number of terms for iterative Fibonacci series: "))

if n <= 0:

    print("Please enter a positive integer.")

else:

    print("Iterative Fibonacci series:")

    print(fibonacci_iterative(n))

n = int(input("Enter the number of terms for recursive Fibonacci series: "))

if n <= 0:

    print("Please enter a positive integer.")

else:

    print("Recursive Fibonacci series:")

    print(fibonacci_recursive(n))

```

Output:

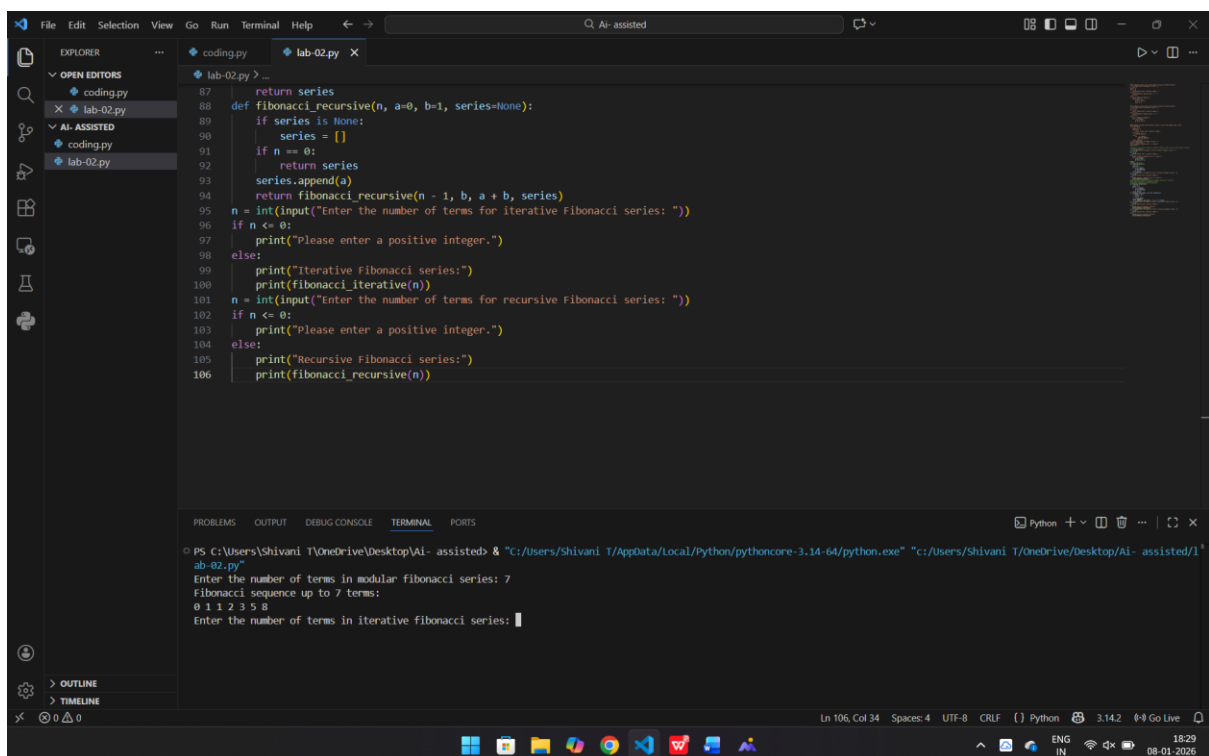
Enter the number of terms in modular fibonacci series: 7

Fibonacci sequence up to 7 terms:

0 1 1 2 3 5 8

Explanation:

1. `fibonacci_iterative(n)` uses a loop: it starts with `a = 0`, `b = 1`, keeps appending `a` to a list, updates `a`, `b = b, a + b`, and finally returns the list.
2. `fibonacci_recursive(n, a, b, series)` builds the same sequence using recursion: it appends `a`, then calls itself with `n-1` until `n` becomes 0.
3. The program asks the user for `n` twice (once for iterative, once for recursive), checks that `n` is positive, and prints each result



The screenshot shows a Visual Studio Code editor with a Python file named `lab-02.py`. The code defines two functions: `fibonacci_iterative` and `fibonacci_recursive`. The `fibonacci_iterative` function uses a loop to generate the Fibonacci sequence, while the `fibonacci_recursive` function uses recursion. The main program prompts the user to enter the number of terms for both series, checks if the input is positive, and prints the results.

```
87     return series
88 def fibonacci_recursive(n, a=0, b=1, series=None):
89     if series is None:
90         series = []
91     if n == 0:
92         return series
93     series.append(a)
94     return fibonacci_recursive(n - 1, b, a + b, series)
95 n = int(input("Enter the number of terms for iterative Fibonacci series: "))
96 if n <= 0:
97     print("Please enter a positive integer.")
98 else:
99     print("Iterative Fibonacci series:")
100    print(fibonacci_iterative(n))
101 n = int(input("Enter the number of terms for recursive Fibonacci series: "))
102 if n <= 0:
103     print("Please enter a positive integer.")
104 else:
105     print("Recursive Fibonacci series:")
106    print(fibonacci_recursive(n))
```

The terminal output shows the execution of the program. It prompts the user to enter the number of terms for the iterative Fibonacci series (7) and the recursive Fibonacci series (7). The output displays the Fibonacci sequence up to 7 terms for both series.

```
PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted> & "C:/Users/Shivani T/AppData/Local/Python/pythoncore-3.14-64/python.exe" "c:/Users/Shivani T/OneDrive/Desktop/Ai- assisted/lab-02.py"
Enter the number of terms in modular fibonacci series: 7
Fibonacci sequence up to 7 terms:
0 1 1 2 3 5 8
Enter the number of terms in iterative fibonacci series: 7
```