

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.

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows files: coding.py, lab-02.py, AI-ASSISTED, coding.py, and lab-02.py.
- Editor View:** Displays the Python code for generating a Fibonacci sequence up to n terms.
- Terminal View:** Shows the execution of the script and the resulting Fibonacci sequence output.
- Bottom Status Bar:** Provides information like file path, encoding, and date.

```

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

```

PS C:\Users\Shivani\OneDrive\Desktop\AI-assisted> & "C:/Users/Shivani/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

```

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.

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows files coding.py and lab-02.py in the Open Editors section. The AI-Assisted section also lists coding.py and lab-02.py.
- Terminal:** Displays the Python command and 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: 9
Fibonacci sequence:
0 1 1 2 3 5 8 13 21
PS C:\Users\Shivani T\OneDrive\Desktop\AI-assisted>
```
- Status Bar:** Shows the file is 30 lines long, has 24 columns, and was last modified on 08-01-2026 at 10:07.

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.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files `coding.py` and `lab-02.py`.
- Editor:** Displays Python code for finding a Fibonacci sequence up to `n` terms. The code includes AI-assisted suggestions for variable names like `a` and `b`, and a function definition for `fibonacci(n)`.
- Terminal:** Shows the command `PS C:\Users\Shivani\OneDrive\Desktop\AI-assisted> & "C:/Users/Shivani/AppData/Local/Python/pythoncore-3.14-64/python.exe" "c:/Users/shivani/OneDrive/Desktop/AI-assisted/lab-02.py"`. It prompts for the number of terms, receives input `7`, and outputs the sequence `[0, 1, 1, 2, 3, 5, 8]`.
- Bottom Status Bar:** Shows file path `C:\Users\Shivani\OneDrive\Desktop\AI-assisted>`, line `Ln 50, Col 14`, spaces `Spaces:4`, encoding `UTF-8`, and date/time `10:39 08-01-2026`.

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

```
File Edit Selection View Go Run Terminal Help <- > Ai-assisted
EXPLORER ... coding.py lab-02.py
OPEN EDITORS coding.py lab-02.py
AI-ASSISTED coding.py lab-02.py
File Edit Selection View Go Run Terminal Help <- > Ai-assisted
EXPLORER ... coding.py lab-02.py
OPEN EDITORS coding.py lab-02.py
AI-ASSISTED coding.py lab-02.py
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))

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 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 code editor interface with two open files: `coding.py` and `lab-02.py`. The `lab-02.py` file contains the following code:

```
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 window below shows the execution of the script and the output of the Fibonacci sequences.

```
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/1\ab-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: 10
Iterative Fibonacci series:
0 1 1 2 3 5 8 13 21 34 55
Recursive Fibonacci series:
0 1 1 2 3 5 8 13 21 34 55
```