

AI ASSISTED CODING

Name: P.Swaran Raj

Roll.No: 2303A52247

Batch: 37

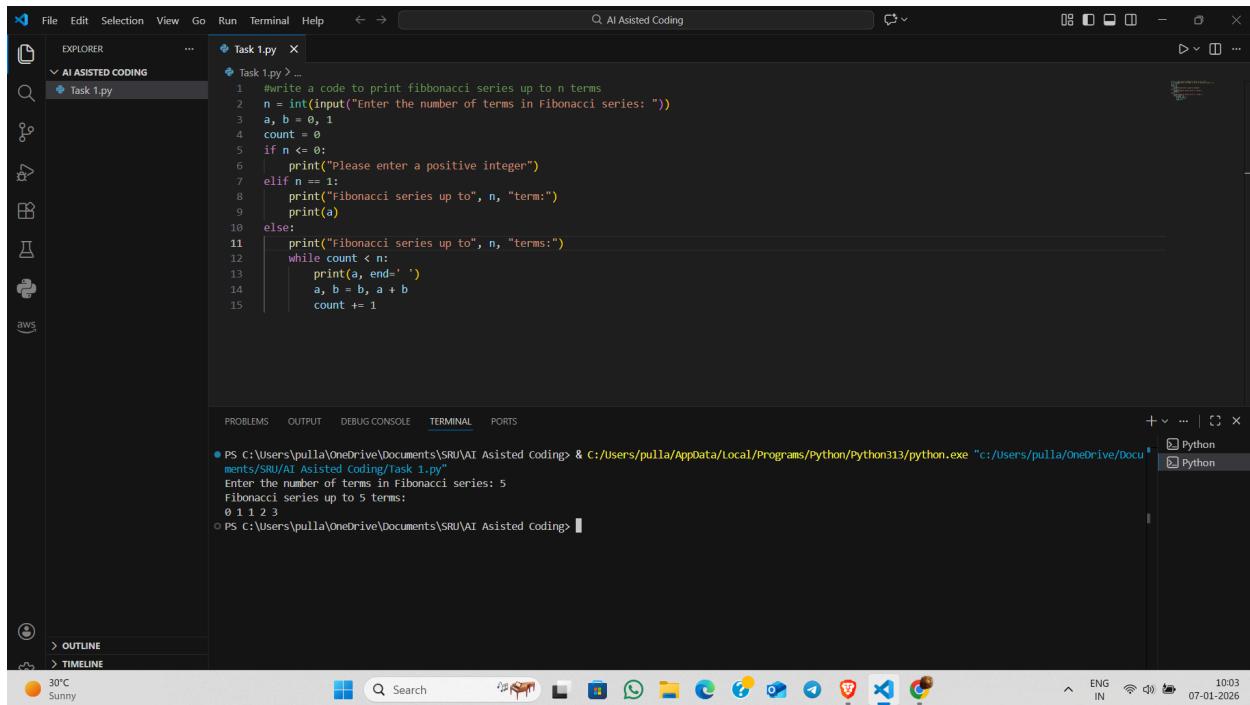
Assignment-1

Task 1- AI-Generated Logic Without Modularization (Fibonacci Sequence Without Functions)

Prompt used:

“#write a code to print fibonacci series up to n terms”

“#write a code to print fibonacci series up to n terms without using functions”



The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a search bar. The left sidebar has sections for Explorer, AI ASSISTED CODING, and Task 1.py. The main editor area displays the following Python code:

```
Task 1.py
Task 1.py ...
1 #write a code to print fibonacci series up to n terms
2 n = int(input("Enter the number of terms in Fibonacci series: "))
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 series up to", n, "term:")
9     print(a)
10 else:
11     print("Fibonacci series up to", n, "terms:")
12     while count < n:
13         print(a, end=" ")
14         a, b = b, a + b
15         count += 1
```

The bottom terminal tab shows the execution of the script and its output:

```
PS C:\Users\pulla\OneDrive\Documents\SRU\AI Assisted Coding> & C:/Users/pulla/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/pulla/OneDrive/Documents\SRU\AI Assisted Coding\Task 1.py"
Enter the number of terms in Fibonacci series: 5
Fibonacci series up to 5 terms:
0 1 1 2 3
```

The status bar at the bottom right indicates the date (07-01-2026), time (10:03), language (ENG IN), and battery level.

```
# write a code to print fibonacci series up to n terms without using functions
n = int(input("Enter the number of terms in Fibonacci series: "))
a, b = 0, 1
count = 0
while count < n:
    print(a, end=" ")
    a, b = b, a + b
    count += 1
```

Explanation:

GitHub Copilot generated a simple loop-based Fibonacci program using inline comments as prompt, The logic is written directly in the main code without any function definition.

Output:

For input 5, the output displayed is: 0 1 1 2 3

Observation:

Copilot correctly understood the prompt and produced working procedural code.

However, the code was less reusable and slightly lengthy.

Task 2: AI Code Optimization & Cleanup (Improving Efficiency)

Prompt used: “ #Optimize this Fibonacci code by simplifying logic and removing redundant variables.”

“#Original code generates Fibonacci series up to n terms”

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

- File Explorer:** Shows a folder named "AI ASISTED CODING" containing "Task 1.py".
- Code Editor:** Displays the following Python code:

```
1 # optimize the code below without using function definition and redundant variables and simplify it
2 # Original code generates Fibonacci series up to n terms
3 n = int(input("Enter the number of terms in Fibonacci series: "))
4 a, b = 0, 1
5 count = 0
6 while count < n:
7     print(a, end=" ")
8     a, b = b, a + b
9     count += 1
```
- Terminal:** Shows the command line output:

```
PS C:\Users\pulla\OneDrive\Documents\AI Asisted Coding> & C:/users/pulla/AppData/Local/Programs/Python/Python313/python
n.exe "c:/Users/pulla/OneDrive/Documents/AI Asisted Coding/task 1.py"
● Enter the number of terms in Fibonacci series: 9
0 1 1 2 3 5 8 13 21
○ PS C:\Users\pulla\OneDrive\Documents\AI Asisted Coding>
```
- Bottom Status Bar:** Shows the date (07-01-2026), time (10:30), and system status (17°C, Mostly sunny).

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

- File Explorer:** Shows a folder named "AI ASISTED CODING" containing "Task 1.py".
- Code Editor:** Displays the following Python code, which is identical to the original but has been optimized by GitHub Copilot:

```
1 # optimize the code below without using function definition and redundant variables
2 # Original code generates Fibonacci series up to n terms
3 n = int(input("Enter the number of terms in Fibonacci series: "))
4 a, b = 0, 1
5 count = 0
6 while count < n:
7     print(a, end=" ")
8     a, b = b, a + b
9     count += 1
```
- Terminal:** Shows the command line output:

```
PS C:\Users\pulla\OneDrive\Documents\AI Asisted Coding> & C:/users/pulla/AppData/Local/Programs/Python/Python313/python
n.exe "c:/Users/pulla/OneDrive/Documents/AI Asisted Coding/task 1.py"
● Enter the number of terms in Fibonacci series: 7
0 1 1 2 3 5 8
○ PS C:\Users\pulla\OneDrive\Documents\AI Asisted Coding>
```
- Bottom Status Bar:** Shows the date (07-01-2026), time (10:28), and system status (17°C, Mostly sunny).

Explanation:

Copilot reduced unnecessary variables and simplified the loop structure. The optimized version is shorter, cleaner, and easier to read.

Output:

For input 7, the output displayed is: 0 1 1 2 3 5 8

Observation:

Optimization improved readability and performance.

The code became more concise while producing the same correct output.

Task 3: Modular Design Using AI Assistance (Fibonacci Using Functions)

Prompt used: "#Optimize this Fibonacci code by simplifying logic and removing redundant variables."

"#Original code generates Fibonacci series up to n terms without using functions"

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

- File Structure:** Explorer sidebar shows "AI ASSISTED CODING" and "Task 1.py".
- Code Editor:** Task 1.py contains the following Python code:

```
# optimize the code below with using function definition and redundant variables and simplify
# Original code generates Fibonacci series up to n terms without using functions
def fibonacci_series(n):
    a, b = 0, 1
    for _ in range(n):
        print(a, end=" ")
        a, b = b, a + b
n = int(input("Enter the number of terms in Fibonacci series: "))
a, b = 0, 1
count = 0
while count < n:
    print(a, end=" ")
    a, b = b, a + b
    count += 1
```
- Terminal:** Shows the command run and the output of the Fibonacci series up to 14 terms.
- Bottom Right Panel:** "Build with Agent" section with instructions and a button.
- Bottom Status Bar:** Includes system icons, language (ENG IN), and date/time (07-01-2026).

Explanation:

Copilot generated a user-defined function to encapsulate Fibonacci logic. The function is called from the main program for better modularity.

Output:

For input 13, the output displayed is:

0 1 1 2 3 5 8 13 21 34 55 89 144

Observation:

Function-based approach improved code clarity and reusability.
This approach is more suitable for larger applications.

Conclusion: GitHub Copilot helped in quickly generating, optimizing, and understanding Fibonacci programs in VS Code, while manual review ensured correctness and better coding practices.