

# Assignment 01

Anyam Akshitha

2303A52453

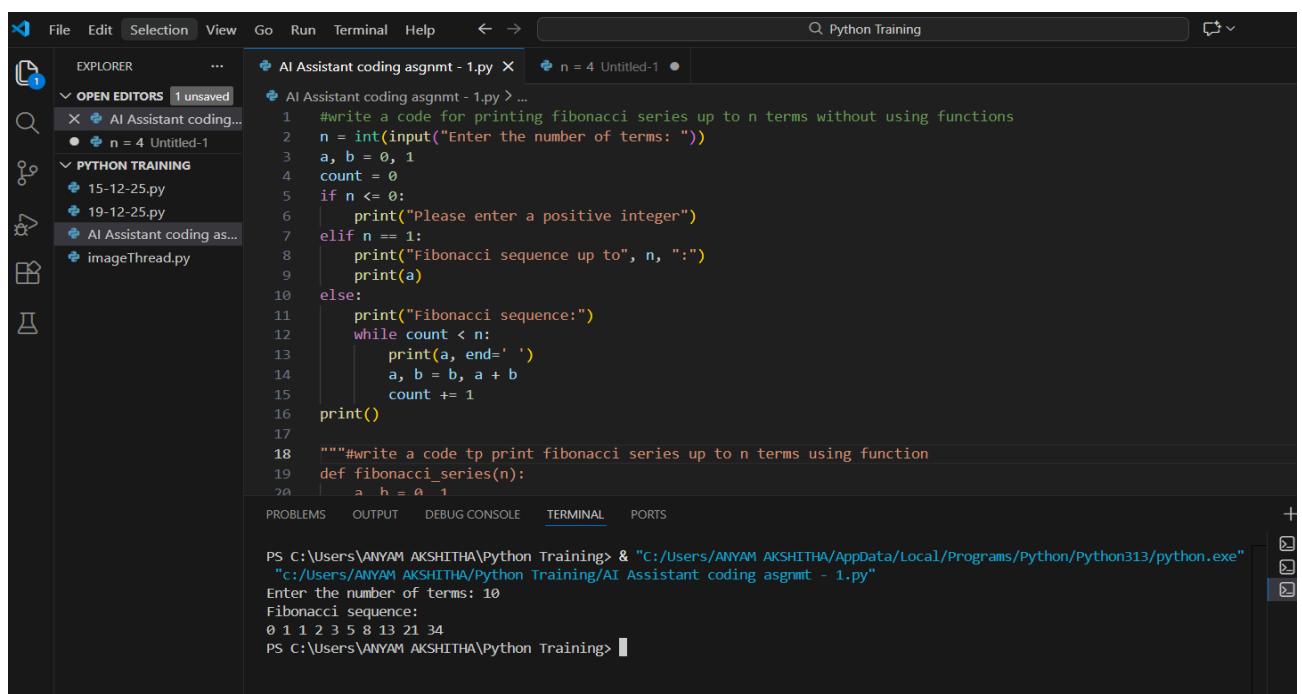
Batch – 42

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

PROMPT:

Write a code for printing fibonacci series up to n terms without using functions

CODE:



The screenshot shows a Python code editor interface with the following details:

- File Explorer:** Shows several Python files: "15-12-25.py", "19-12-25.py", "AI Assistant coding as...", and "imageThread.py".
- Open Editors:** One file is open: "AI Assistant coding asgnmt - 1.py".
- Code Content:**

```
#write a code for printing fibonacci series up to n terms without using 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 up to", n, ":")
    print(a)
else:
    print("Fibonacci sequence:")
    while count < n:
        print(a, end=' ')
        a, b = b, a + b
        count += 1
print()
"""
#write a code tp print fibonacci series up to n terms using function
def fibonacci_series(n):
    a = 0
    b = 1
```
- Terminal:** Shows the command line output:

```
PS C:\Users\ANYAM AKSHITHA\Python Training> & "C:/Users/ANYAM AKSHITHA/AppData/Local/Programs/Python/Python313/python.exe"
"c:/Users/ANYAM AKSHITHA/Python Training/AI Assistant coding asgnmt - 1.py"
Enter the number of terms: 10
Fibonacci sequence:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANYAM AKSHITHA\Python Training>
```

## OUTPUT :

The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with 'OPEN EDITORS' and 'PYTHON TRAINING' sections. The main area displays a Python script named 'AI Assistant coding asgmt - 1.py'. The code prints a Fibonacci sequence up to n terms. Below the code, the terminal window shows the command run and the resulting Fibonacci sequence output.

```
1 #write a code for printing fibonacci series up to n terms without using 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 up to", 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     print()
17
18 """#write a code tp print fibonacci series up to n terms using function
19 def fibonacci_series(n):
20     a, b = 0, 1
```

PS C:\Users\ANYAM AKSHITHA\Python Training> & "C:/Users/ANYAM AKSHITHA/AppData/Local/Programs/Python/Python313/python.exe"  
"c:/Users/ANYAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"  
Enter the number of terms: 10  
Fibonacci sequence:  
0 1 1 2 3 5 8 13 21 34  
PS C:\Users\ANYAM AKSHITHA\Python Training>

## JUSTIFICATION:

By completing this task , we have printed the required fibonacci series up to n terms without modularization. In this code we are printing the series with only one for loop.

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

### PROMPT:

Optimized version of Fibonacci series up to n terms without using functions

### CODE:

The screenshot shows the Visual Studio Code interface with the Python Training extension installed. In the Explorer sidebar, there are several files: 15-12-25.py, 19-12-25.py, AI Assistant coding as... (which is currently open), and imageThread.py. The terminal window shows the execution of a script named 'AI Assistant coding asgmt - 1.py'. The script prompts for the number of terms and prints the Fibonacci sequence up to that number. The output in the terminal is:

```
PS C:\Users\ANVYAM AKSHITHA\Python Training> & "C:/Users/ANVYAM AKSHITHA/appData/Local/Programs/Python/Python313/python.exe" "c:/Users/ANVYAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"
Enter the number of terms: 10
Fibonacci sequence:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANVYAM AKSHITHA\Python Training> & "C:/Users/ANVYAM AKSHITHA/appData/Local/Programs/Python/Python313/python.exe" "c:/Users/ANVYAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"
Enter the number of terms: 10
Fibonacci sequence:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANVYAM AKSHITHA\Python Training>
```

The right-hand sidebar features the AI Assistant interface with a message bubble icon and the text 'Ask about your code'. It also includes a note that AI responses may be inaccurate and a link to 'Generate Agent Instructions'.

## OUTPUT :

This screenshot is identical to the one above, showing the same code editor setup, terminal output, and AI Assistant sidebar. The terminal shows the execution of 'AI Assistant coding asgmt - 1.py' twice, both times prompting for 10 terms and printing the Fibonacci sequence from 0 to 34.

## JUSTIFICATION:

By completing this task , we have got the optimized version of the previous code , as we can see the code which we have got the in task 1 is already optimized one. Optimization of the code is very much important, as it reduces the time complexity and space complexity of the code, for the code which has more time complexity takes more time to run(slower) and which has more space complexity will utilize maximum space. So for completing our task efficiently , we need to Optimize our code.

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

## PROMPT:

Optimized version of Fibonacci series up to n terms using functions

## CODE:

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a search bar for "Python Training". The left sidebar has icons for File, Recent, Find, Replace, and a Chat section titled "CHAT". The main editor area displays Python code for generating a Fibonacci series:

```
19 #Optimized version of Fibonacci series up to n terms using functions
20 def fibonacci_series(n):
21     a, b = 0, 1
22     series = []
23     for _ in range(n):
24         print(a, end=" ")
25         a, b = b, a + b
26     n = int(input("Enter the number of terms in Fibonacci series: "))
27     print("Fibonacci series:")
28     fibonacci_series(n)
```

The terminal below shows the execution of the script:

```
PS C:\Users\ANYAM AKSHITHA\Python Training> python.exe "c:/Users/ANYAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"
10
PS C:\Users\ANYAM AKSHITHA\Python Training> 8 "c:/Users/ANYAM AKSHITHA/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/ANYAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"
Enter the number of terms in Fibonacci series: 10
Fibonacci series:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANYAM AKSHITHA\Python Training>
```

A floating "Ask about your code" panel is visible on the right, with a message: "Ask about your code". Below it, a note says "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase." The bottom status bar shows "Ln 23, Col 23" and "Spaces: 4".

## OUTPUT:

The screenshot shows a Python code editor interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** Python Training.
- Code Area:** A script named `fibonacci_series.py` containing optimized code for generating a Fibonacci series using functions.

```
19 #Optimized version of Fibonacci series up to n terms using functions
20 def fibonacci_series(n):
21     a, b = 0, 1
22     series = []
23     for _ in range(n):
24         print(a, end=" ")
25         a, b = b, a + b
26     n = int(input("Enter the number of terms in Fibonacci series: "))
27     print("\nFibonacci series:")
28     fibonacci_series(n)
```

- Terminal Tab:** Shows the command `python fibonacci_series.py` being run in a Windows terminal, outputting the first 10 terms of the Fibonacci series.

```
PS C:\Users\ANVAM AKSHITHA\Python Training> python fibonacci_series.py
10
PS C:\Users\ANVAM AKSHITHA\Python Training> python fibonacci_series.py
Enter the number of terms in Fibonacci series: 10
Fibonacci series:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANVAM AKSHITHA\Python Training>
```

- Bottom Status Bar:** Lines 23, Columns 23, Spaces: 4, UTF-8, CRLF, Python, 3.13.7, Iniciar Live.
- System Icons:** Volume, Battery, Network, Weather (26°C, Sunny), ENG IN, and a QR code.
- Right Sidebar:** CHAT tab, a message icon, and a "Ask about your code" button.

## JUSTIFICATION:

By completing this task , we are able to print the Fibonacci series up to n numbers using functions by using the above code. At first we are calling the user-defined function using function calling statement and then our function runs. In our function the main for loop runs and prints the output.

## Task 4: Comparative Analysis – Procedural vs Modular Fibonacci Code

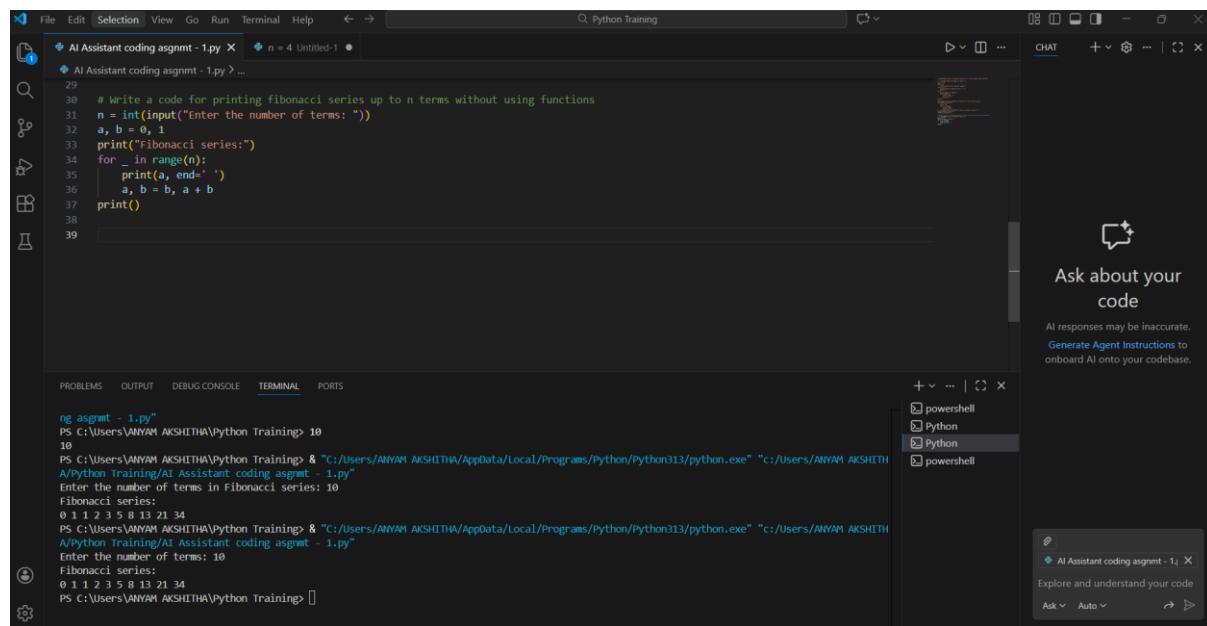
### PROMPTS:

Procedural : Write a code for printing fibonacci series up to n terms without using functions

Modular : Optimized version of Fibonacci series up to n terms using functions

### CODE:

#### Procedural:



A screenshot of the Visual Studio Code (VS Code) interface. The top bar shows the title "Python Training". The left sidebar has icons for file operations like Open, Save, and Find. The main editor area contains Python code for generating a Fibonacci series. The bottom status bar shows the path "C:\Users\ANYAM AKSHITHA\Python Training> 10" and the terminal tab is active. The terminal window shows the execution of the script and the resulting Fibonacci sequence from 0 to 34. A floating "Ask about your code" window is visible on the right side of the interface.

```
29
30 # Write a code for printing fibonacci series up to n terms without using functions
31 n = int(input("Enter the number of terms: "))
32 a, b = 0, 1
33 print("Fibonacci series:")
34 for _ in range(n):
35     print(a, end=" ")
36     a, b = b, a + b
37 print()
```

```
ng asgmt - 1.py"
PS C:\Users\ANYAM AKSHITHA\Python Training> 10
10
PS C:\Users\ANYAM AKSHITHA\Python Training> & "C:/Users/ANYAM AKSHITHA/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/ANYAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"
AI/Python Training/AI Assistant coding asgmt - 1.py"
Enter the number of terms in Fibonacci series: 10
Fibonacci series:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANYAM AKSHITHA\Python Training> & "C:/Users/ANYAM AKSHITHA/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/ANYAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"
AI/Python Training/AI Assistant coding asgmt - 1.py"
Enter the number of terms: 10
Fibonacci series:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANYAM AKSHITHA\Python Training> []
```

#### Modular:

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a search bar for "Python Training". The left sidebar has icons for file operations like Open, Save, and Undo. The main editor area contains Python code for generating a Fibonacci series up to n terms:

```
File Edit Selection View Go Run Terminal Help < > Q Python Training
AI Assistant coding asgnmt - 1.py X n = 4 Untitled-1 ...
AI Assistant coding asgnmt - 1.py > fibonacci_series
36     a, b = b, a + b
37     print()"""
38
39 # optimized version of Fibonacci series up to n terms using functions
40 def fibonacci_series(n):
41     a, b = 0, 1
42     for _ in range(n):
43         print(a, end=" ")
44         a, b = b, a + b
45 n = int(input("Enter the number of terms in Fibonacci series: "))
46 print("Fibonacci series:")
47 fibonacci_series(n)
48
49
```

The status bar at the bottom shows "In 41, Col 16 Spaces: 4 UTF-8 CRLF () Python 3.13.7 Iniciar Go Live". A floating "Ask about your code" callout box is visible on the right, along with a message: "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase." The bottom navigation bar includes icons for Home, Search, and various file types.

## OUTPUT:

## Procedural:

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

- File Explorer:** Shows two files: "AI Assistant coding asgnmt - 1.py" and "n = Untitled-1".
- Code Editor:** Displays Python code for generating a Fibonacci series up to n terms. The code uses a for loop and variable assignments.
- Terminal:** Shows the execution of the script and its output. The output includes the command run, the path to the Python executable, the script name, user input for the number of terms, and the resulting Fibonacci series.
- Bottom Right Panel:** An AI Assistant sidebar titled "Ask about your code" with a "CHAT" button and a "Generate Agent Instructions" link.
- Bottom Left Panel:** A "PROBLEMS" panel showing no errors.
- Bottom Center:** A dropdown menu for "New File" with options like "powershell", "Python", and "Python file".

## Modular:

```

File Edit Selection View Go Run Terminal Help < > Python Training
AI Assistant coding asgmt - 1.py X n = 4 Untitled-1
AI Assistant coding asgmt - 1.py > fibonacci_series
36 |     a, b = b, a + b
37 |     print()
38 |
39 # Optimized version of Fibonacci series up to n terms using functions
40 def fibonacci_series():
41     a, b = 0, 1
42     for _ in range(n):
43         print(a, end=' ')
44         a, b = b, a + b
45     n = int(input("Enter the number of terms in Fibonacci series: "))
46     print("Fibonacci series:")
47     fibonacci_series(n)
48
49
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Enter the number of terms in Fibonacci series: 10
Fibonacci series:
Traceback (most recent call last):
  File "c:/Users/ANVAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py", line 47, in <module>
    fibonacci_series()
               ~~~~~
  File "c:/Users/ANVAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py", line 41, in fibonacci_series
    a, b = 0,
           ~~~
ValueError: not enough values to unpack (expected 2, got 1)
PS C:/Users/ANVAM AKSHITHA/Python Training> & "c:/Users/ANVAM AKSHITHA/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/ANVAM AKSHITHA/Python Training/AI Assistant coding asgmt - 1.py"
Enter the number of terms in Fibonacci series: 10
Fibonacci series:
0 1 1 2 3 5 8 13 21 34
PS C:/Users/ANVAM AKSHITHA/Python Training> []

```

Ask about your code  
AI responses may be inaccurate.  
Generate Agent Instructions to onboard AI onto your codebase.

+ v ... | x  
powershell  
Python  
Python  
powershell

AI Assistant coding asgmt - 1.py  
Explore and understand your code  
Ask Auto ▶

In 41, Col 16 Spaces:4 UTF-8 CR/LF { } Python 3.13.7 Iniciar Go Live 26°C Sunny ENG IN 13:53 08-01-2026

## JUSTIFICATION:

By this task , we are able to find the difference between Procedural(without using functions) and Modular(with using functions). The main use of function is

- Reusability of the code
- Easy to Debug
- Code Clarity • Suitable for large systems

By observing, we can analyze that using modular method is a better and clean aproch

## Task 5: AI-Generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches for Fibonacci Series)

### PROMPTS:

Iterative approach: fibonacci series up to n terms using iterative approach

Recursive approach : fibonacci series up to n terms using recursive approach

### CODE:

Recursive approach:

A screenshot of the Visual Studio Code (VS Code) interface. The title bar says "Python Training". The left sidebar shows two tabs: "AI Assistant coding asgmt - 1.py" and "n = 4 Untitled-1". The main editor area contains Python code for an iterative Fibonacci series generator. The terminal at the bottom shows the output of running the script, which asks for the number of terms (10) and prints the series [0, 1, 2, 3, 5, 8, 13, 21, 34]. The status bar at the bottom right shows the date (08-01-2026), time (13:56), and weather (26°C, Sunny).

```
50
51 #Fibonacci series up to n terms using iterative approach
52 def fibonacci_series(n):
53     a, b = 0, 1
54     for _ in range(n):
55         print(a, end=' ')
56         a, b = b, a + b
57 n = int(input("Enter the number of terms in Fibonacci series: "))
58 print("Fibonacci series:")
59 fibonacci_series(n)
```

Enter the number of terms in Fibonacci series: 10  
Fibonacci series:  
0 1 1 2 3 5 8 13 21 34  
PS C:\Users\ANYAM AKSHITHA\Python Training>

## Iterative approach:

A screenshot of the Visual Studio Code (VS Code) interface. The title bar says "Python Training". The left sidebar shows two tabs: "AI Assistant coding asgmt - 1.py" and "fibonacci\_recursive". The main editor area contains Python code for a recursive Fibonacci series generator. The terminal at the bottom shows the output of running the script, which asks for the number of terms (10) and prints the series [0, 1, 2, 3, 5, 8, 13, 21, 34]. The status bar at the bottom right shows the date (08-01-2026), time (14:00), and weather (26°C, Sunny).

```
61
62 # fibonacci series up to n terms using recursive approach
63 def fibonacci_recursive(n):
64     if n <= 0:
65         return []
66     elif n == 1:
67         return [0]
68     elif n == 2:
69         return [0, 1]
70     else:
71         series = fibonacci_recursive(n - 1)
72         series.append(series[-1] + series[-2])
73     return series
74 n = int(input("Enter the number of terms in Fibonacci series: "))
75 print("Fibonacci series:")
76 print(' '.join(map(str, fibonacci_recursive(n))))
```

Enter the number of terms in Fibonacci series: 10  
Fibonacci series:  
Traceback (most recent call last):  
File "c:\users\anyam akshitha\python training\ai assistant coding asgmt - 1.py", line 76, in <module>  
 print(' '.join(map(str, fibonacci\_recursive(n))))  
 ^~~~~~  
File "c:\users\anyam akshitha\python training\ai assistant coding asgmt - 1.py", line 71, in fibonacci\_recursive  
 series = fibonacci(n - 1)  
 ^~~~~~  
NameError: name 'fibonacci' is not defined  
PS C:\Users\ANYAM AKSHITHA\Python Training & "c:/users/ANYAM AKSHITHA/AppData/Local/Programs/Python/Python313/python.exe" "c:/users/ANYAM AKSHITHA/Python Training\AI Assistant coding asgmt - 1.py"  
Enter the number of terms in Fibonacci series: 10  
Fibonacci series:  
0 1 1 2 3 5 8 13 21 34  
PS C:\Users\ANYAM AKSHITHA\Python Training>

## OUTPUT:

### Recursive approach:

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with the following details:

- File Explorer:** On the left, showing a file named "AI Assistant coding asgnmt - 1.py".
- Code Editor:** The main area displays Python code for generating a Fibonacci series:

```
50 #fibonacci series up to n terms using iterative approach
51 def fibonacci_series(n):
52     a, b = 0, 1
53     for _ in range(n):
54         print(a, end=' ')
55         a, b = b, a + b
56 n = int(input("Enter the number of terms in Fibonacci series: "))
57 print("Fibonacci series:")
58 fibonacci_series(n)
```
- Terminal:** At the bottom, showing the output of running the script:

```
Enter the number of terms in Fibonacci series: 10
Fibonacci series:
0 1 1 2 3 5 8 13 21 34
PS C:\Users\ANVYAM AKSHITHA\Python Training>
```
- AI Assistant:** A sidebar on the right titled "Ask about your code" with a message "Ask about your code". Below it, a note says "AI responses may be inaccurate." and "Generate Agent Instructions to onboard AI onto your codebase."
- PowerShell Context Menu:** A context menu is open over the terminal tab, listing options: powershell, Python, and PowerShell.

## Iterative approach:

## **JUSTIFICATION:**

The iterative Fibonacci approach is usually the better choice because it's faster, uses very little memory, and works well even when the number of terms is large. Recursive Fibonacci, on the other hand, makes many function calls, which slows things down and uses extra memory. For bigger inputs, it can even

crash due to recursion limits. That's why, in real-world programs, iteration is generally preferred over recursion.