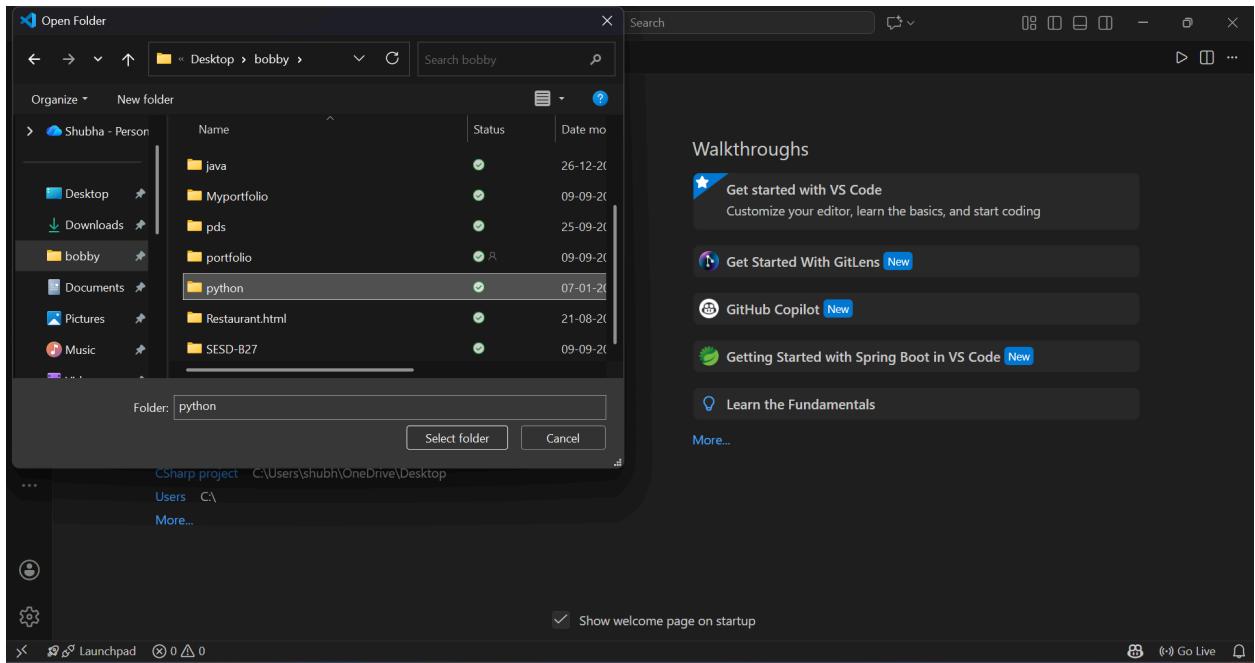


Lab Assignment 1.3

P.Naga Shiva Chaitanya (2303A51945)

The screenshot shows the Visual Studio Code Marketplace interface. The search bar at the top has "python" typed into it. On the left sidebar, there's a list of extensions under "EXTENSIONS: MARKETPLACE". One extension is highlighted: "Python" by Microsoft, which has a rating of 5 stars and over 198,510,521 downloads. The main panel displays the "Python extension for Visual Studio Code" details, including its logo (a blue and yellow Python icon), developer information (Microsoft, ms-python.python), version (2025.28.1), and installation statistics. To the right, there's a "CHAT" section with a message about summing two numbers in Python, followed by code snippets for a script and a reusable function.

This screenshot shows the same VS Code Marketplace interface, but the search bar now has "git" typed into it. The "GitHub Copilot" extension by GitHub is highlighted, showing a rating of 5 stars and 66,427,573 downloads. The main panel displays the "GitHub Copilot - Your AI peer programmer" details, including its logo (a white circular icon with a stylized head), developer information (GitHub, github.copilot), version (1.388.0), and installation statistics. Similar to the Python extension, it includes a "CHAT" section with a message about summing two numbers in Python, followed by code snippets for a script and a reusable function.



Task 1: AI-Generated Logic Without Modularization (Procedural Fibonacci) :

Fibonacci sequence

A screenshot of the Visual Studio Code interface, specifically focusing on the Python development environment. The Explorer sidebar shows a 'PYTHON' folder containing 'Fibonacci.py' and 'sum.py'. The 'Fibonacci.py' file is open in the editor, displaying the following code:

```
# Fibonacci sequence
n = int(input("Enter the number of terms: "))
a, b = 0, 1
count = 0
while count < n:
    print(a, end=' ')
    a, b = b, a + b
    count += 1
```

The 'TERMINAL' tab is active, showing the output of running the script: "Enter the number of terms: 4" followed by the printed sequence "0 1 1 2". The status bar at the bottom indicates the code is on line 3, column 12, in Python mode, and the port is 5500.

Task 2: AI Code Optimization & Cleanup :

Optimize this Fibonacci code

Simplify variable usage

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

- File Explorer:** Shows two files: `Fibonacci.py` and `sum.py`. `Fibonacci.py` is the active file.
- Code Editor:** Displays the following Python code:

```
1
2  # Optimize this Fibonacci code
3  # Simplify variable usage
4  n = int(input("Enter the number of terms: "))
5  a, b = 0, 1
6  for _ in range(n):
7      print(a, end=' ')
8      a, b = b, a + b
```
- Terminal:** Shows the command `python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"` being run, followed by the output of the Fibonacci sequence: `0 1 1 2`.
- Status Bar:** Shows the current file is `Fibonacci.py`, line 3, column 26, spaces: 4, encoding: UTF-8, LF, Python 3.14.0, port 5500, and Prettier.

Task 3: Modular Design Using AI Assistance (Function-Based Fibonacci) :

Write a Python function to generate Fibonacci sequence up to n

Use meaningful comments

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

- File Explorer:** Shows two files: `Fibonacci.py` and `sum.py`. `Fibonacci.py` is the active file.
- Code Editor:** Displays the following Python code:

```
1  # Write a Python function to generate Fibonacci sequence up to n
2  # Use meaningful comments
3  def fibonacci_sequence(n):
4      """Generate Fibonacci sequence up to n terms."""
5      sequence = []
6      a, b = 0, 1
7      for _ in range(n):
8          sequence.append(a)
9          a, b = b, a + b
10     return sequence
```
- Terminal:** Shows the command `python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"` being run, followed by the output of the Fibonacci sequence: `0 1 1 2`.
- Status Bar:** Shows the current file is `Fibonacci.py`, line 1, column 1, spaces: 4, encoding: UTF-8, LF, Python 3.14.0, port 5500, and Prettier.

Task 4: Comparative Analysis – Procedural vs Modular Code

Criteria	Without Functions	With Functions
Code Clarity	Lower	Higher
Reusability	No	Yes
Debugging	Harder	Easier
Scalability	Poor	Excellent
Suitable for Large Systems	No	Yes

Task 5: Iterative vs Recursive Fibonacci (AI-Generated):

Generate Fibonacci using iterative approach

Generate Fibonacci using recursive approach

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

- File Explorer:** Shows a folder named "PYTHON" containing "Fibonacci.py" and "sum.py".
- Code Editor:** Displays the content of "Fibonacci.py".

```
1 # Generate Fibonacci using recursive approach
2
3 def fibonacci_recursive(n):
4     if n <= 0:
5         return 0
6     elif n == 1:
7         return 1
8     else:
9         return fibonacci_recursive(n - 1) + fibonacci_recursive(n - 2)
```
- Terminal:** Shows the output of running the script.

```
C:\Users\shubh\OneDrive\Desktop\bobby\python>python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"
Enter the number of terms: 4
0 1 1 2
C:\Users\shubh\OneDrive\Desktop\bobby\python>python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"
Enter the number of terms: 4
0 1 1 2
C:\Users\shubh\OneDrive\Desktop\bobby\python>
```
- Status Bar:** Shows the file path as "C:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py", line 1, column 1, and other standard status bar information.