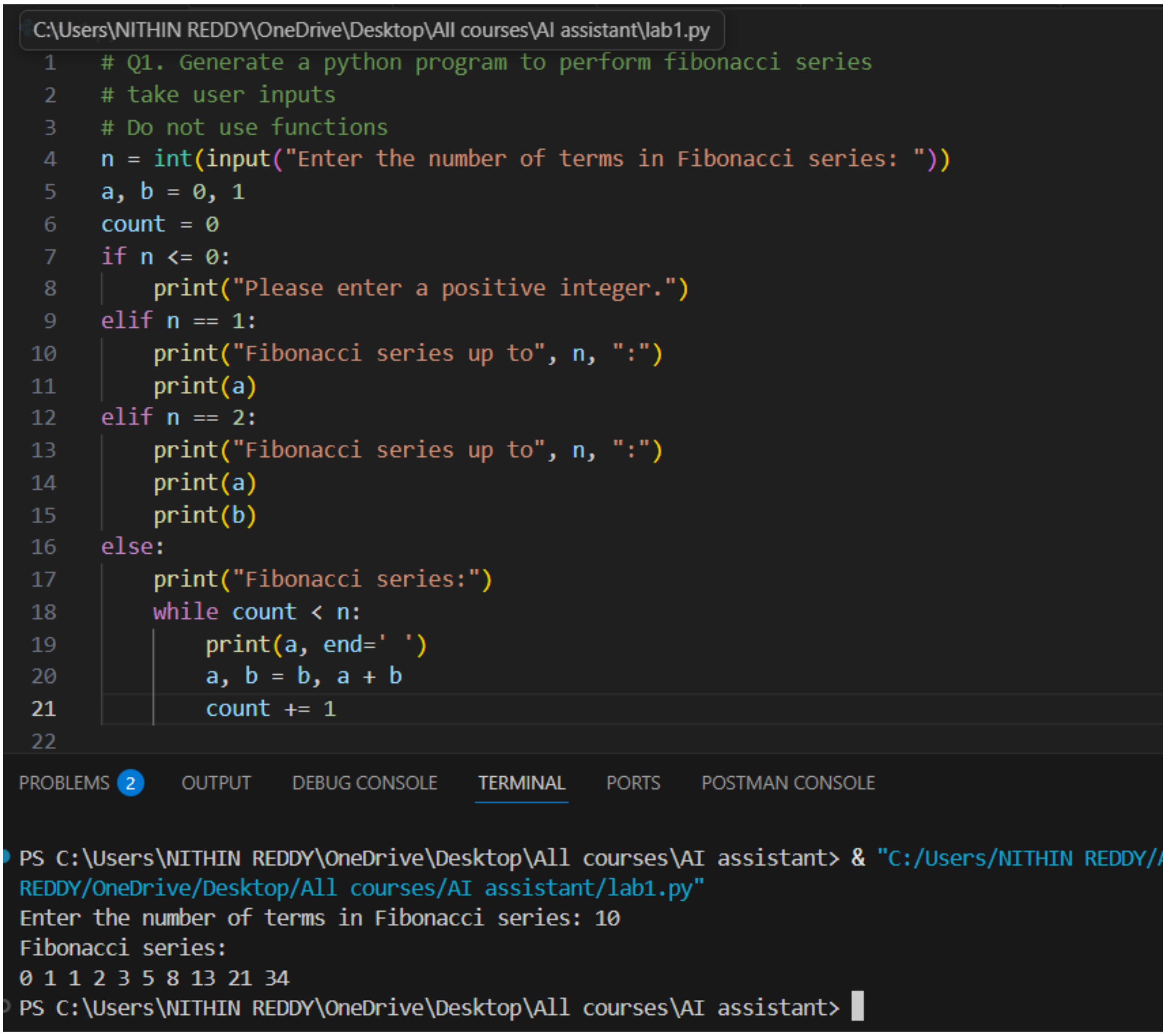
AI Assisted Coding

Assignment - 1

Ch. Nithin Reddy || 2303A51550 || Batch:- 8

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

Code:

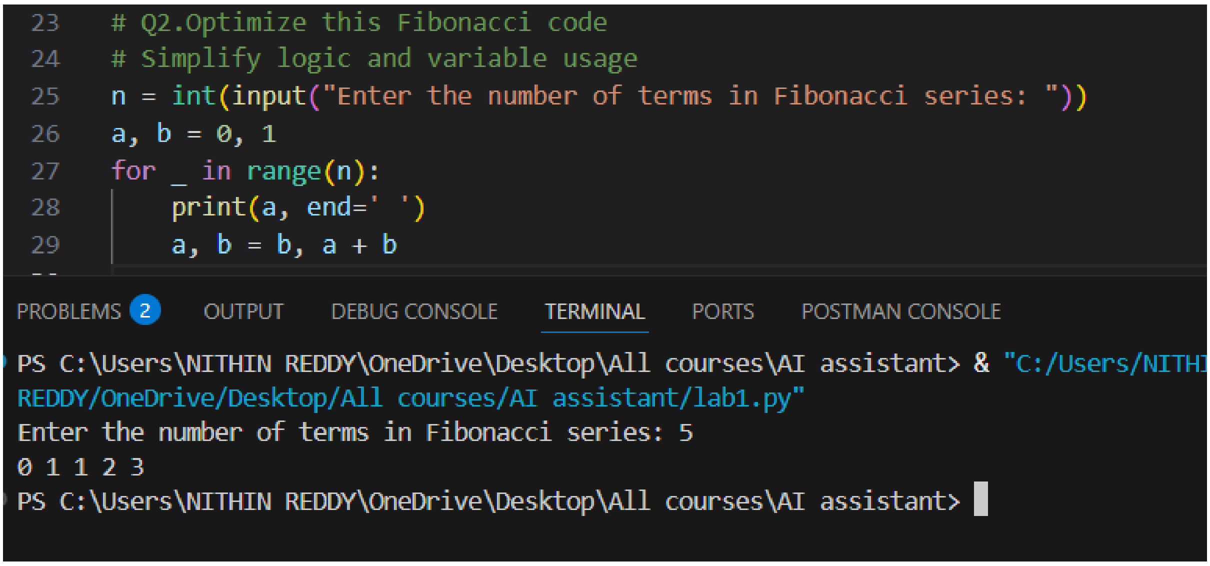

C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant\lab1.py
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
POSTMAN CONSOLE

```python
# Q1. Generate a python program to perform fibonacci series
# take user inputs
# Do not use functions
n = int(input("Enter the number of terms in Fibonacci series: "))
a, b = 0, 1
count = 0
if n <= 0:
    print("Please enter a positive integer.")
elif n == 1:
    print("Fibonacci series up to", n, ":")
    print(a)
elif n == 2:
    print("Fibonacci series up to", n, ":")
    print(a)
    print(b)
else:
    print("Fibonacci series:")
    while count < n:
        print(a, end=' ')
        a, b = b, a + b
        count += 1

```

```
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant> & "C:/Users/NITHIN REDDY/A
REDDY/OneDrive/Desktop/All courses/AI assistant/lab1.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\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant> 
```

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

Code:

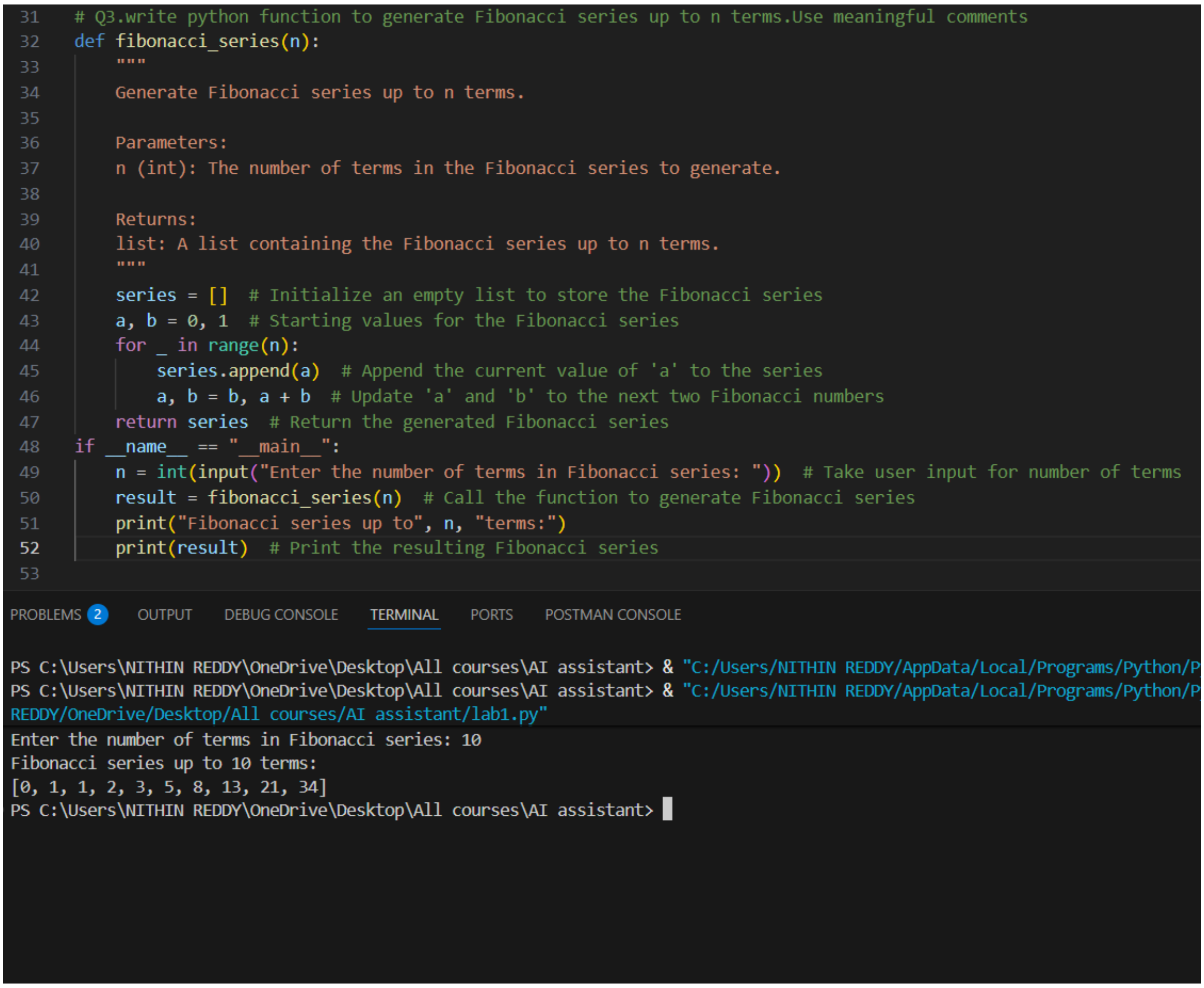


```python
# Q2.Optimize this Fibonacci code
# Simplify logic and variable usage
n = int(input("Enter the number of terms in Fibonacci series: "))
a, b = 0, 1
for _ in range(n):
    print(a, end=' ')
    a, b = b, a + b
```

```
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant> & "C:/Users/NITHI
REDDY/OneDrive/Desktop/All courses/AI assistant/lab1.py"
Enter the number of terms in Fibonacci series: 5
0 1 1 2 3
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant>
```

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

Code:

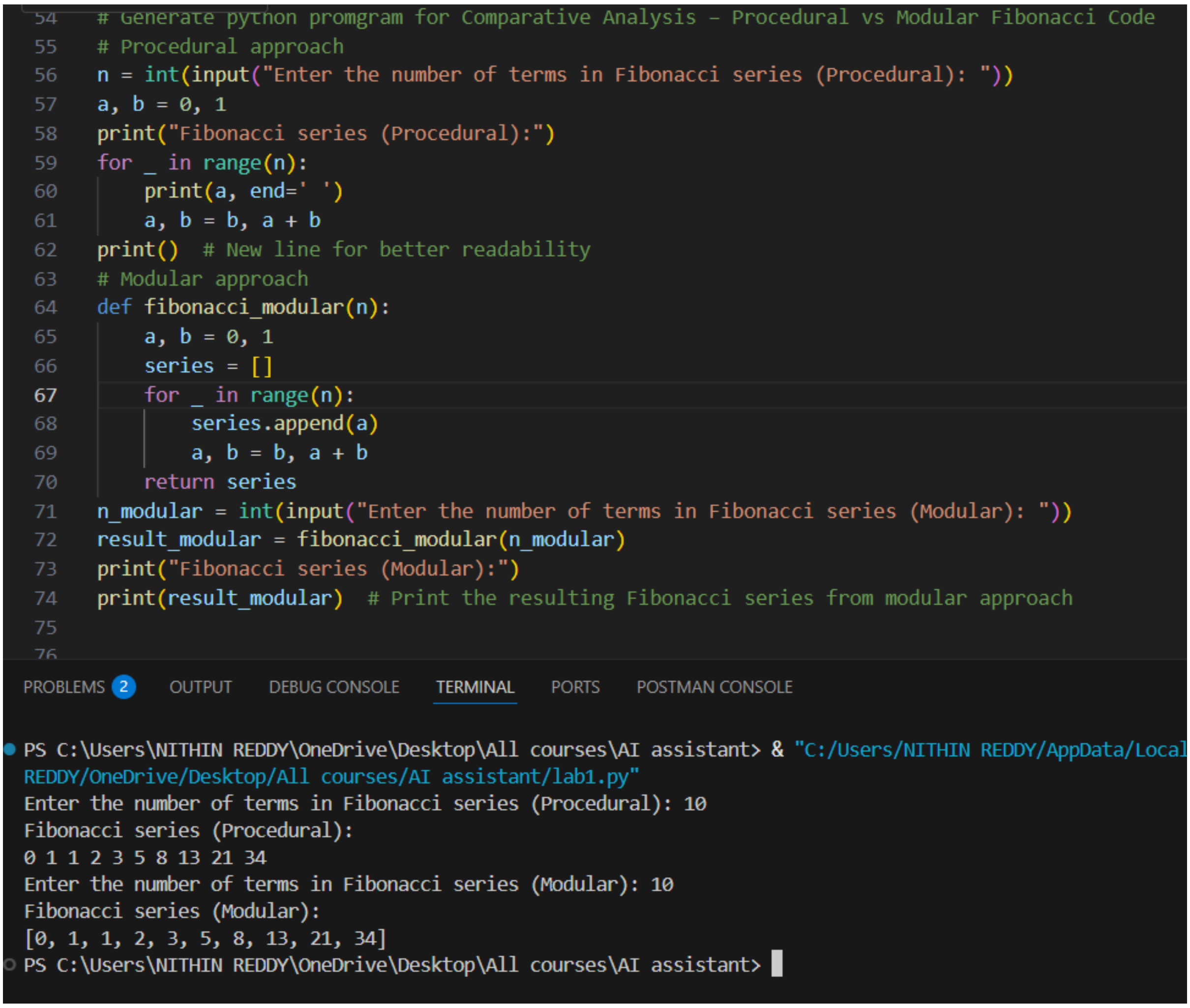


```python
# Q3.write python function to generate Fibonacci series up to n terms.Use meaningful comments
def fibonacci_series(n):
    """
    Generate Fibonacci series up to n terms.

    Parameters:
    n (int): The number of terms in the Fibonacci series to generate.

    Returns:
    list: A list containing the Fibonacci series up to n terms.
    """
    series = []  # Initialize an empty list to store the Fibonacci series
    a, b = 0, 1  # Starting values for the Fibonacci series
    for _ in range(n):
        series.append(a)  # Append the current value of 'a' to the series
        a, b = b, a + b  # Update 'a' and 'b' to the next two Fibonacci numbers
    return series  # Return the generated Fibonacci series
if __name__ == "__main__":
    n = int(input("Enter the number of terms in Fibonacci series: "))  # Take user input for number of terms
    result = fibonacci_series(n)  # Call the function to generate Fibonacci series
    print("Fibonacci series up to", n, "terms:")
    print(result)  # Print the resulting Fibonacci series

```

```
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant> & "C:/Users/NITHIN REDDY/AppData/Local/Programs/Python/P
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant> & "C:/Users/NITHIN REDDY/AppData/Local/Programs/Python/P
REDDY/OneDrive/Desktop/All courses/AI assistant/lab1.py"
Enter the number of terms in Fibonacci series: 10
Fibonacci series up to 10 terms:
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant>
```

Task 4: Comparative Analysis – Procedural vs Modular Fibonacci Code

Code:

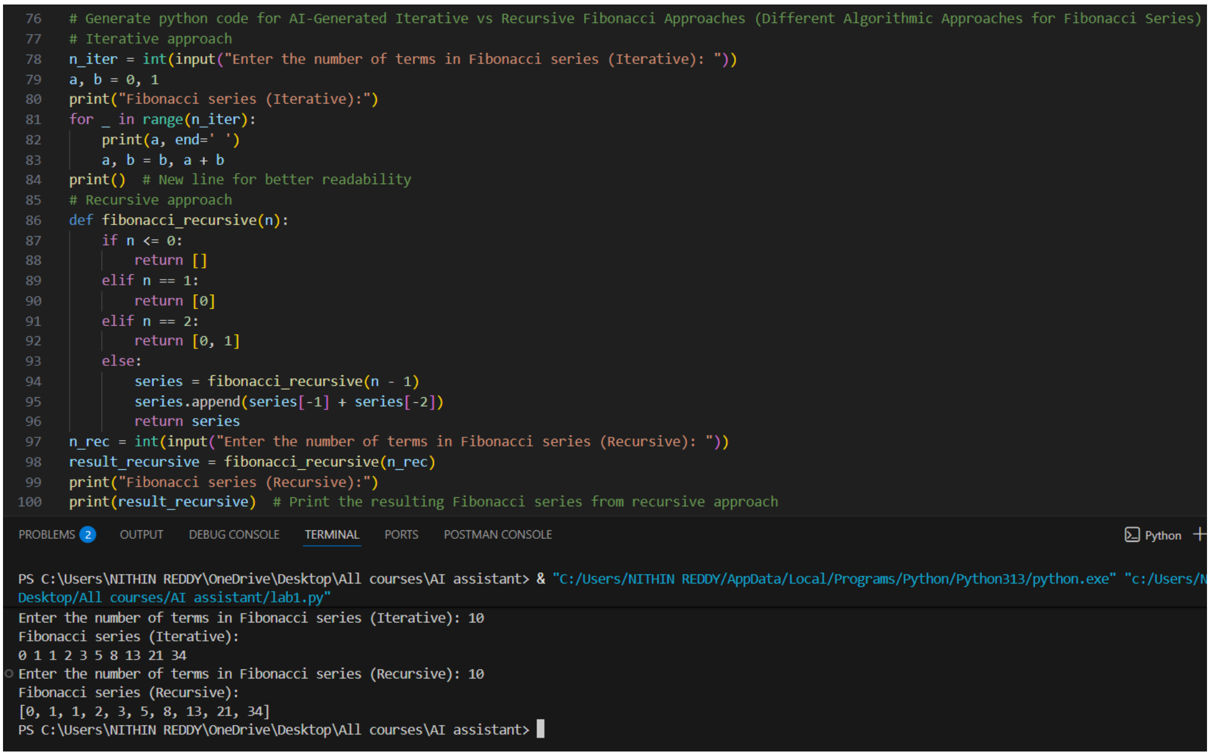


```python
# Generate python promgram for Comparative Analysis – Procedural vs Modular Fibonacci Code
# Procedural approach
n = int(input("Enter the number of terms in Fibonacci series (Procedural): "))
a, b = 0, 1
print("Fibonacci series (Procedural):")
for _ in range(n):
    print(a, end=' ')
    a, b = b, a + b
print()  # New line for better readability
# Modular approach
def fibonacci_modular(n):
    a, b = 0, 1
    series = []
    for _ in range(n):
        series.append(a)
        a, b = b, a + b
    return series
n_modular = int(input("Enter the number of terms in Fibonacci series (Modular): "))
result_modular = fibonacci_modular(n_modular)
print("Fibonacci series (Modular):")
print(result_modular)  # Print the resulting Fibonacci series from modular approach
```

```
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant> & "C:/Users/NITHIN REDDY/AppData/Local
REDDY/OneDrive/Desktop/All courses/AI assistant/lab1.py"
Enter the number of terms in Fibonacci series (Procedural): 10
Fibonacci series (Procedural):
0 1 1 2 3 5 8 13 21 34
Enter the number of terms in Fibonacci series (Modular): 10
Fibonacci series (Modular):
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant>
```

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

Code:



```python
# Generate python code for AI-Generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches for Fibonacci Series)
# Iterative approach
n_iter = int(input("Enter the number of terms in Fibonacci series (Iterative): "))
a, b = 0, 1
print("Fibonacci series (Iterative):")
for _ in range(n_iter):
    print(a, end=' ')
    a, b = b, a + b
print()  # New line for better readability
# Recursive approach
def fibonacci_recursive(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    else:
        series = fibonacci_recursive(n - 1)
        series.append(series[-1] + series[-2])
        return series
n_rec = int(input("Enter the number of terms in Fibonacci series (Recursive): "))
result_recursive = fibonacci_recursive(n_rec)
print("Fibonacci series (Recursive):")
print(result_recursive)  # Print the resulting Fibonacci series from recursive approach
```

```
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant> & "C:/Users/NITHIN REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/N
Desktop/All courses/AI assistant/lab1.py"
Enter the number of terms in Fibonacci series (Iterative): 10
Fibonacci series (Iterative):
0 1 1 2 3 5 8 13 21 34
Enter the number of terms in Fibonacci series (Recursive): 10
Fibonacci series (Recursive):
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
PS C:\Users\NITHIN REDDY\OneDrive\Desktop\All courses\AI assistant>
```