

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

- Editor:** The main editor window displays the file `ASSIGNMENT-6.py`. The code is:

```
# AI-assisted class generation for stude.py
1 # Generate python code to print fibonacci series upto n terms without using functions
2 n = int(input("Enter the number of terms: "))
3 a, b = 0, 1
4 for i in range(n):
5     print(a, end=" ")
6     a, b = b, a + b
7 def print_fibonacci(n):
8     a, b = 0, 1
9     for i in range(n):
10        print(a, end=" ")
11        a, b = b, a + b
```
- Terminal:** The terminal window shows the command `python3 "Users/ronithreddyrevuri/Desktop/AI Asstd Coding/ASSIGNMENT-6.py"` being run, followed by the output of the generated Fibonacci sequence.
- Output:** The output panel shows the command and its execution.
- Problems:** The problems panel shows no errors.
- Right Panel:** The sidebar includes a "Next steps (pick one)" section with options like scaffolding files or walking through another language, and a "Tell me which option you want (scaffold files now / different language / just explain more), and I'll do it." input field.

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

- Editor:** The main editor window displays the file `ASSIGNMENT-6.py`. The code is:

```
# Optimize the python code to print fibonacci series upto n terms
# Generate the code to print fibonacci series upto n terms
def fibonacci(n):
    a, b = 0, 1
    for i in range(n):
        print(a, end=" ")
        a, b = b, a + b
n = int(input("Enter the number of terms: "))
fibonacci(n)
```
- Terminal:** The terminal window shows the command `python3 "Users/ronithreddyrevuri/Desktop/AI Asstd Coding/ASSIGNMENT-6.py"` being run, followed by a `KeyboardInterrupt` error message.
- Output:** The output panel shows the command and its execution, including the error message.
- Problems:** The problems panel shows no errors.
- Right Panel:** The sidebar includes a "Next steps (pick one)" section with options like scaffolding files or walking through another language, and a "Tell me which option you want (scaffold files now / different language / just explain more), and I'll do it." input field.

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

- File Explorer:** Shows "OPEN EDITORS" with "Welcome" and "# AI-assisted class generation for stude.py".
- Code Editor:** Displays Python code for generating a Fibonacci series:

```
# Optimize the python code to print fibonacci series upto n terms
# Generate the code to print fibonacci series upto n terms
# Simplified version of the code to print fibonacci series

n = int(input("Enter the number of terms: "))
def fibonacci(n):
    a, b = 0, 1
    for i in range(n):
        print(a, end=" ")
        a, b = b, a + b
    print()
fibonacci(n)
```
- Terminal:** Shows the output of running the script:

```
Enter the number of terms: 6
0 1 1 2 3 5
```
- Bottom Status Bar:** Shows "Ln 12, Col 13 Spaces: 4 UTF-8 LF Python" and "Python 3.13.9".

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

- EXPLORER**: Shows "OPEN EDITORS" with "Welcome" and "# AI-assisted class generation for stude.py".
- WORKBENCH**: Shows "ASSIGNMENT-6.PY" and "OUTLINE".
- TERMINAL**: Shows the command line output for running the script.
- EDITOR**: Displays the Python code for generating a Fibonacci series, with syntax highlighting for variables like `n` and `a, b`.
- PROBLEMS**: Shows no errors or warnings.
- OUTPUT**: Shows the terminal output.
- DEBUG CONSOLE**: Shows no output.
- TUTORIAL**: Shows a sidebar titled "AI-assisted class generation for stude.py" with a "Next steps (pick one)" section containing two items:

 - I can scaffold the files (README.md, hello.py, tests/, requirements.txt, .gitignore) in your workspace now.
 - Or walk through another language (JavaScript/TypeScript, Java, etc.) if you prefer.

- NOTES**: Shows a note: "Tell me which option you want (scaffold files now / different language / just explain more), and I'll do it."
- COMMAND PALETTES**: Shows "Add Context...", "new", "#new", and "#new" again.
- STATUS BAR**: Shows "Ln 14, Col 1" and "Python 3.13.9".

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

- EXPLORER**: Shows the project structure with files like `Welcome`, `# AI-assisted class generation for stude.py`, `ASSIGNMENT-6.PY`, `OUTLINE`, and `TIMELINE`.
- EDITOR**: Displays the Python code for generating a Fibonacci series. The code uses a recursive function `fibonacci(n)` and prints terms up to `n`. A comment at the top specifies optimizing the code to print the series up to `n` terms.
- TERMINAL**: Shows the terminal output of running the script. It prompts for the number of terms and prints the series. The output includes several error messages from the AI-assistant about Python version mismatch and forgetting to activate `.venv`.
- RIGHT SIDE BAR**: Includes a "Next steps (pick one)" section with options like scaffolding files or walking through another language, and a "Tell me which option you want" dropdown.

```
# # AI-assisted class generation for stude.py > ...
1 # Optimize the python code to print fibonacci series upto n terms
2 # Generate the code to print fibonacci series upto n terms
3 # Simplified version of the code to print fibonacci series
4 # Generate the fibonacci series using defined function
5 # Generate code for iterative fibonacci function
6
7 def fibonacci(n):
8     a, b = 0, 1
9     for _ in range(n):
10         print(a, end=" ")
11         a, b = b, a + b
12         print()
13
14 n = int(input("Enter the number of terms: "))
15 fibonacci(n)
16
```

```
I Asstd Coding/ASSIGNMENT-6.py/# AI-assisted class generation for stude.py"
Enter the number of terms: 5
0 1 1 2 3
ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py % /usr/local/bin/python3
on3 "/Users/ronithreddyrevuri/Desktop/AI Asstd Coding/ASSIGNMENT-6.py/# AI-
assisted class generation for stude.py"
Enter the number of terms: ~Ctraceback (most recent call last):
File "/Users/ronithreddyrevuri/Desktop/AI Asstd Coding/ASSIGNMENT-6.py/# AI-assisted class generation for stude.
py", line 13, in <module>
KeyboardInterrupt
```

```
ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py % /usr/local/bin/python3 "/Users/ronithreddyrevuri/Desktop/AI
Asstd Coding/ASSIGNMENT-6.py/# AI-assisted class generation for stude.py"
Enter the number of terms: 4
0 1 1 2
ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py %
```

This screenshot is nearly identical to the first one, showing the same code editor, terminal output, and right-side bar. The only difference is the line numbers in the code editor, which have changed from 16 to 17.

```
# # AI-assisted class generation for stude.py > ...
1 # Simplified version of the code to print fibonacci series
2 # Generate the fibonacci series using defined function
3 # Generate code for iterative fibonacci function
4 # Generate code for recursive fibonacci function
5
6 def fibonacci(n):
7     if n <= 1:
8         return n
9     return fibonacci(n - 1) + fibonacci(n - 2)
10
11 n = int(input("Enter the number of terms: "))
12
13 for i in range(n):
14     print(fibonacci(i), end=" ")
15
16
```

```
ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py % /usr/local/bin/python3 "/Users/ronithreddyrevuri/Desktop/AI
Asstd Coding/ASSIGNMENT-6.py/# AI-assisted class generation for stude.py"
Enter the number of terms: ~Ctraceback (most recent call last):
File "/Users/ronithreddyrevuri/Desktop/AI Asstd Coding/ASSIGNMENT-6.py/# AI-assisted class generation for stude.
py", line 13, in <module>
n = int(input("Enter the number of terms: "))
KeyboardInterrupt
```

```
ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py % /usr/local/bin/python3 "/Users/ronithreddyrevuri/Desktop/AI
Asstd Coding/ASSIGNMENT-6.py/# AI-assisted class generation for stude.py"
Enter the number of terms: 6
0 1 1 2 3 5
ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py %
```

TIME AND SPACE COMPLEXITY COMPARISON

ABSTRACT

THE FIBONACCI SERIES IS ACHIEVED THROUGH AN ITERATIVE AND RECURSIVE APPROACH.

THE ITERATIVE SOLUTION FOLLOWS THE LOOP MECHANISM WITH A TIME COMPLEXITY OF $O(N)$ AND SPACE COMPLEXITY OF $O(1)$. IT IS VERY FAST WITH MINIMAL MEMORY REQUIREMENT, MAKING THE SOLUTION EFFICIENT FOR LARGE NUMBERS TOO.

TO BE SPECIFIC, IN THE RECURSIVE METHOD, THE FUNCTION IS CALLED MANY TIMES. AS A RESULT, THE TIME COMPLEXITY IS $O(2^n)$ AND THE SPACE COMPLEXITY IS $O(N)$.

CONCLUSION

THE ITERATIVE METHOD IS EFFICIENT AND CAN BE USED WITH LARGER NUMBERS, WHILE THE RECURSIVE METHOD IS INEFFICIENT AND IS NOT RECOMMENDED FOR LARGER VALUES OF N .