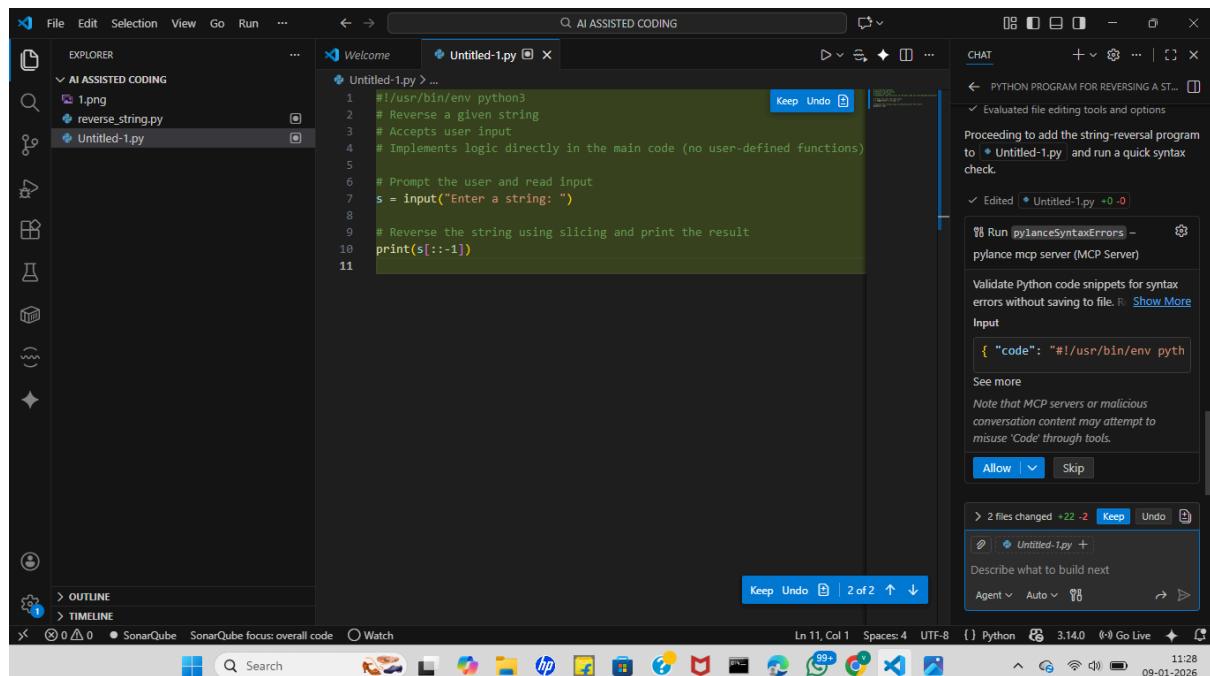
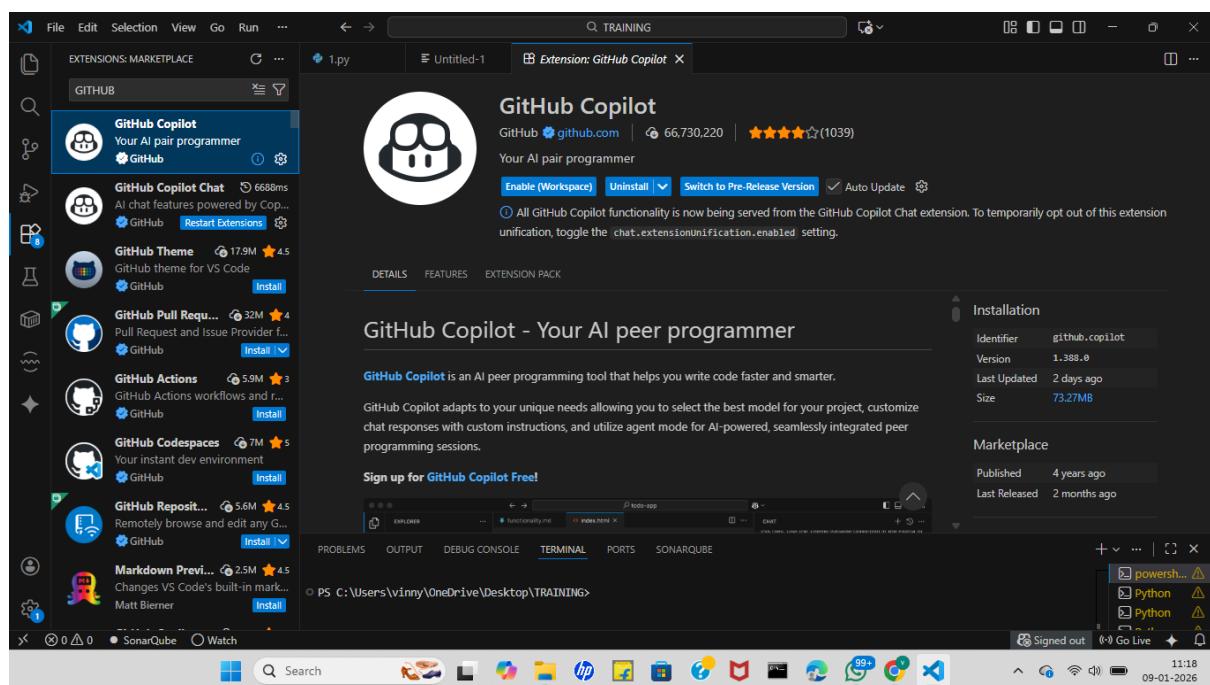


Batch 03  
2303A51449

### Task Description 1:

Use GitHub Copilot to generate a Python program that:

- Reverses a given string
- Accepts user input
- Implements the logic directly in the main code
- Does not use any user-defined functions



```

1  #!/usr/bin/env python3
2  # Reverse a given string
3  # Accepts user input
4  # Implements logic directly in the main code (no user-defined functions)
5
6  # Prompt the user and read input
7  s = input("Enter a string: ")
8
9  # Reverse the string using slicing and print the result
10 print(s[::-1])
11

```

PS C:\Users\vinny\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/vinny/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/vinny/OneDrive/Desktop/AI ASSISTED CODING/reverse\_string.py"  
Enter a string: ADITHYA  
AYHTIDA

PS C:\Users\vinny\OneDrive\Desktop\AI ASSISTED CODING>

Enter a string: ADITHYA  
AYHTIDA

## Task Description 2:

Examine the Copilot-generated code from Task 1 and improve it by:

- Removing unnecessary variables
- Simplifying loop or indexing logic
- Improving readability
- Use Copilot prompts like:
  - “Simplify this string reversal code”
  - “Improve readability and efficiency”

```

1  #!/usr/bin/env python3
2  # Reverse a given string
3  # Accepts user input
4  # Implements logic directly in the main code (no user-defined functions)
5
6  # Prompt the user and read input
7  s = input("Enter a string: ")
8
9  # Reverse the string using slicing and print the result
10 print(s[::-1])
11

```

PS C:\Users\vinny\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/vinny/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/vinny/OneDrive/Desktop/AI ASSISTED CODING/reverse\_string.py"  
Enter a string: ADITHYA  
AYHTIDA

PS C:\Users\vinny\OneDrive\Desktop\AI ASSISTED CODING>

- What I changed
- Added a minimal string-reversal program directly into • Untitled-1.py that:
    - Prompts for user input,
    - Reverses the string with slicing (`s[::-1]`) in main code (no user-defined functions),
    - Prints the reversed string.

### How to run

Open PowerShell in the workspace and run:  
python • Untitled-1.py

Then type a string when prompted — it will

reverse\_string.py +  
Describe what to build next

Agent ▾ Auto ▾

3.14.0 (↻ Go Live) 11:29 09-01-2026

```

PS C:\Users\vinny\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/vinny/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/vinny/OneDrive/Desktop/AI ASSISTED CODING/reverse_string.py"
Enter a string: ADITHYA
AYHTIDA
PS C:\Users\vinny\OneDrive\Desktop\AI ASSISTED CODING>

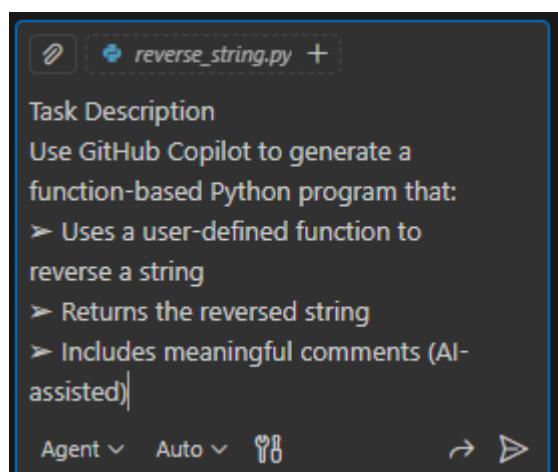
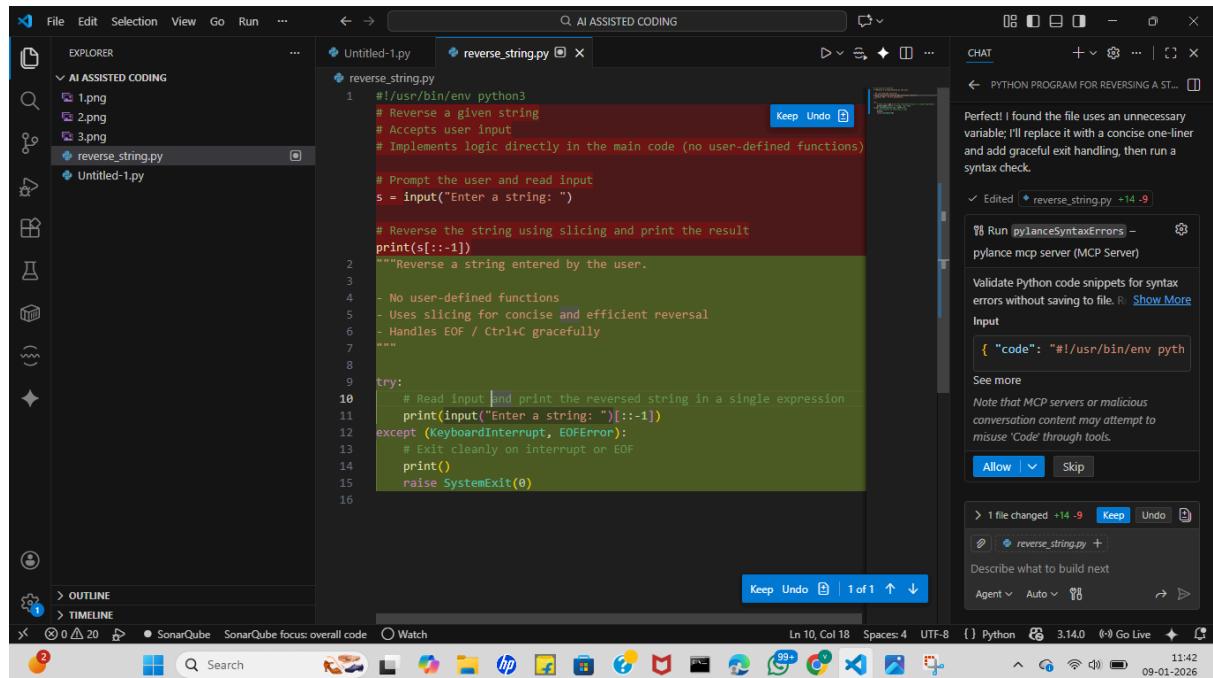
```

OPTIMIZE THE CODE,SIMPLIOFT THE LOGIC,MAKE IT MORE READABLE

### Task Description 3:

Use GitHub Copilot to generate a function-based Python program that:

- Uses a user-defined function to reverse a string
- Returns the reversed string
- Includes meaningful comments (AI-assisted)



Enter a string: ADITHYA  
AYHTIDA

#### TASK DESCRIPTION 4:

Compare the Copilot-generated programs:

- Without functions (Task 1)
- With functions (Task 3)

Analyze them based on:

- Code clarity
- Reusability
- Debugging ease
- Suitability for large-scale applications

#### OUTPUT:

Quick summary table 		
Criterion	reverse_string.py (no-function)	reverse_string_function.py (function)
Code clarity	 Very concise but minimal context	 Explicit and documented
Reusability	 Low — hard to import/test	 High — straightforward to import/use
Debugging ease	 Harder to unit-test / step-debug	 Easy to unit-test and inspect
Large-scale suitability	 Not suitable for scale	 Well-suited for scale & extension

## SHORT ANALYTICAL REPORT:

The function-based version ( reverse\_string\_function.py ) is clearer, more reusable, and easier to debug, making it the better foundation for production.

The one-liner ( reverse\_string.py ) is concise and fine for quick interactive use, but not suitable for scale.

Analysis (concise)

Code clarity

Function: High — docstrings, type hints, explicit behavior.

One-liner: Low–Medium — very short but lacks context and docs.

#### Reusability

Function: High — importable and can be reused in other code.

One-liner: Poor — logic embedded in main flow.

#### Debugging ease :

Function: Easy — pure function, straightforward unit tests.

One-liner: Harder — one expression, hard to step through or test.

#### Suitability for large-scale apps :

Function: Good (with tests, CI, validation, and monitoring).

One-liner: Unsuitable — no API, tests, or automation hooks.

## Task Description 5:

Prompt GitHub Copilot to generate:

- A loop-based string reversal approach
- A built-in / slicing-based string reversal approach

The image shows two side-by-side screenshots of the Visual Studio Code (VS Code) interface, both displaying the same Python code for reversing strings. The top screenshot shows a loop-based implementation, and the bottom screenshot shows a slicing-based implementation. Both screenshots include an AI-assisted coding sidebar on the right.

**Top Screenshot (Loop-based Implementation):**

```
#!/usr/bin/env python3
"""
Demonstrate two string-reversal methods.
Includes:
- loop-based implementation
- slicing-based implementation
Both functions return the reversed string.
from typing import List

def reverse_loop(s: str) -> str:
    """Reverse a string using an explicit loop.
    Implementation:
    - Build a list of characters by walking the string backwards, then join
    - Time: O(n), Space: O(n).
    """
    chars: List[str] = []
    for i in range(len(s) - 1, -1, -1):
        chars.append(s[i])
    return ''.join(chars)
```

**Bottom Screenshot (Slicing-based Implementation):**

```
def reverse_slice(s: str) -> str:
    """Reverse a string using Python slicing (concise and fast).
    This is the idiomatic approach in Python and is implemented in C for
    efficiency, so it's typically the fastest for general use.
    """
    return s[::-1]

if __name__ == "__main__":
    import sys

    try:
        if len(sys.argv) > 1:
            text = " ".join(sys.argv[1:])
        else:
            text = input("Enter a string to reverse: ")

        print("Loop-based: ", reverse_loop(text))
        print("Slicing-based: ", reverse_slice(text))
    except (EOFError, KeyboardInterrupt):
        print()
        raise SystemExit(0)
```

