

## AI ASSISTED CODING

2303A52315

BATCH 45

### Task 1: AI-Generated Logic Without Modularization (String Reversal Without Functions)

#### Scenario

You are developing a basic text-processing utility for a messaging application.

#### Task Description

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

#### ❖ Expected Output

- Correct reversed string
- Screenshots showing Copilot-generated code suggestions
- Sample inputs and outputs

The screenshot shows the Visual Studio Code (VS Code) interface. The code editor has a dark theme and displays a Python file named '1.5.py'. The code implements string reversal logic directly within the main function. The terminal below shows the execution of the script and its output. A sidebar on the right contains AI-related features like 'Build with Agent' and a message about AI responses being inaccurate.

```
# take input from user
text = input("Enter a string to reverse: ")

# reverse logic directly in main code
reversed_text = ""

index = len(text) - 1
while index >= 0:
    reversed_text += text[index]
    index -= 1

# display result
print("Reversed string:", reversed_text)
```

PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"  
Enter a string to reverse: sun  
Reversed string: nus

### Task 2: Efficiency & Logic Optimization (Readability Improvement)

#### Scenario

The code will be reviewed by other developers.

## Task Description

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”

Hint:

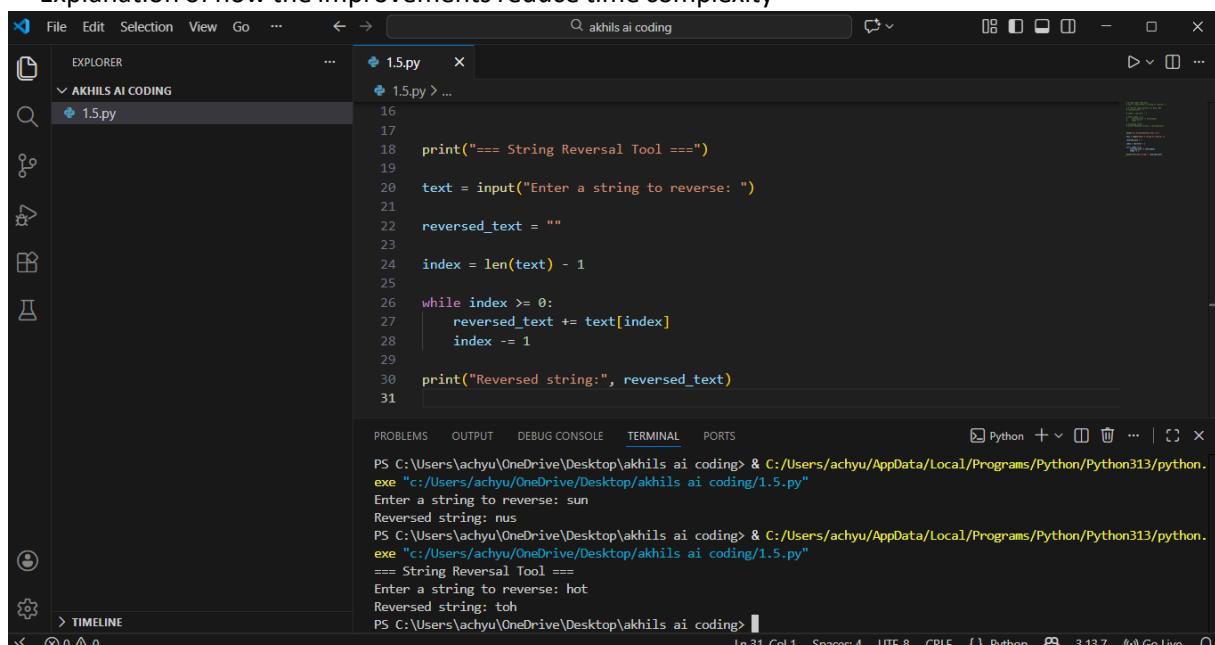
Prompt Copilot with phrases like

“optimize this code”, “simplify logic”, or “make it more readable”

### ❖ Expected Output

Original and optimized code versions

### ➤ Explanation of how the improvements reduce time complexity



```
1.5.py > ...
16
17
18 print("== String Reversal Tool ==")
19
20 text = input("Enter a string to reverse: ")
21
22 reversed_text = ""
23
24 index = len(text) - 1
25
26 while index >= 0:
27     reversed_text += text[index]
28     index -= 1
29
30 print("Reversed string:", reversed_text)
31

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
Enter a string to reverse: sun
Reversed string: nus
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
== String Reversal Tool ==
Enter a string to reverse: hot
Reversed string: toh
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding>
```

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

### Scenario

The string reversal logic is needed in multiple parts of an application.

### Task Description

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)

Expected Output

Correct function-based implementation

Screenshots documenting Copilot's function generation

Sample test cases and outputs

```
1.5.py
33 def reverse_string(value: str) -> str:
34     characters = []
35
36     for i in range(len(value) - 1, -1, -1):
37         characters.append(value[i])
38
39     return "".join(characters)
40
41
42 print("== Modular String Reversal Tool ==")
43
44 user_text = input("Enter a string to reverse: ")
45
46 result = reverse_string(user_text)
47
48 print("Reversed string:", result)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
== String Reversal Tool ==
Enter a string to reverse: hot
Reversed string: toh
PS C:/Users/achyu/OneDrive/Desktop/akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
== Modular String Reversal Tool ==
Enter a string to reverse: kol
Reversed string: lok
PS C:/Users/achyu/OneDrive/Desktop/akhils ai coding>
```

## Task 4: Comparative Analysis – Procedural vs Modular Approach (With vs Without Functions)

### Without Functions)

#### ❖ Scenario

You are asked to justify design choices during a code review.

#### ❖ Task Description

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

- ❖ Expected Output

Comparison table or short analytical report

```

File Edit Selection View Go ... akhils ai coding
EXPLORER AKHILS AI CODING ...
1.5.py ...
1.5.py > ...
49
50
51     print("== String Reversal Tool ==")
52
53     text = input("Enter a string to reverse: ")
54
55     reversed_text = ""
56
57     for ch in text:
58         reversed_text = ch + reversed_text
59
60     print("Reversed string:", reversed_text)
61
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
== Modular String Reversal Tool ==
Enter a string to reverse: kol
Reversed string: lok
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
== String Reversal Tool ==
Enter a string to reverse: eok
Reversed string: koe
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding>

```

### Task 5: AI-Generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches to String Reversal)

- ❖ Scenario

Your mentor wants to evaluate how AI handles alternative logic paths.

- ❖ Task Description

Prompt GitHub Copilot to generate:

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

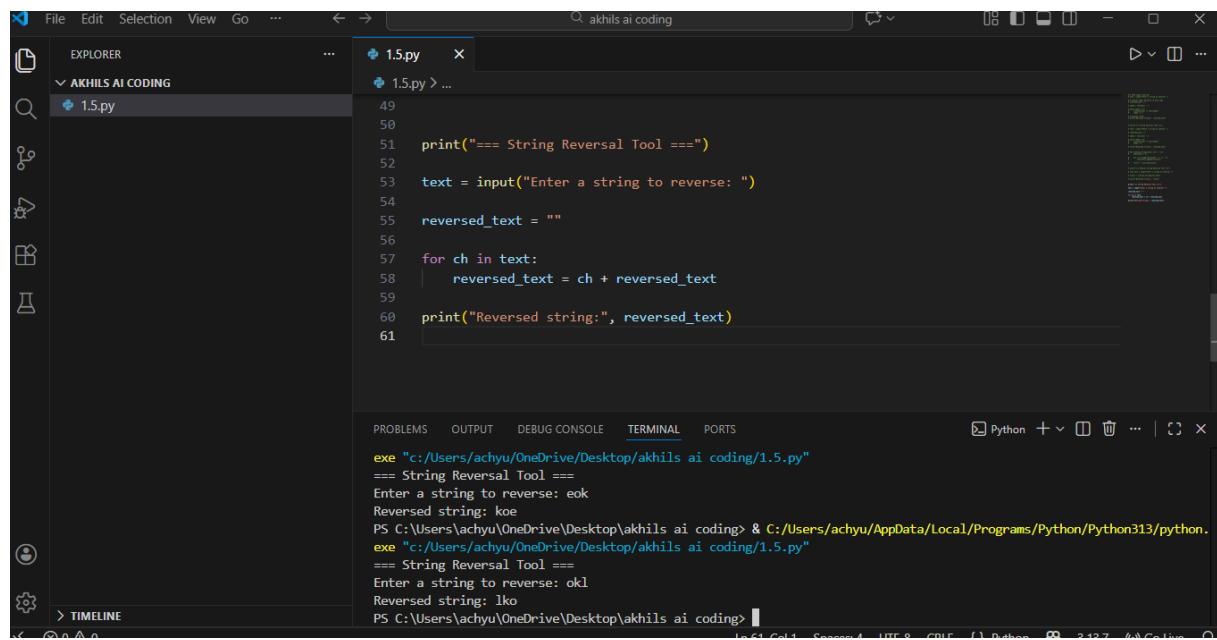
- ❖ Expected Output

- Two correct implementations
- Comparison discussing:
  - Execution flow
  - Time complexity
  - Performance for large inputs
  - When each approach is appropriate

Note: Report should be submitted as a word document for all tasks in a

single document with prompts, comments & code explanation, and output

and if required, screenshots



A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows the 'EXPLORER' view with a folder named 'AKHILS AI CODING' containing a file '1.5.py'. The main editor area displays the following Python code:

```
49
50
51 print("== String Reversal Tool ==")
52
53 text = input("Enter a string to reverse: ")
54
55 reversed_text = ""
56
57 for ch in text:
58     reversed_text = ch + reversed_text
59
60 print("Reversed string:", reversed_text)
61
```

The bottom right corner of the editor shows a small screenshot of the terminal window. Below the editor is the 'TERMINAL' tab, which is active. The terminal window shows the execution of the script and its output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
== String Reversal Tool ==
Enter a string to reverse: eok
Reversed string: oke
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/1.5.py"
== String Reversal Tool ==
Enter a string to reverse: okl
Reversed string: lko
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding>
```

The status bar at the bottom indicates the terminal has 51 rows, 4 columns, and is using UTF-8 encoding.