ASSIGNMENT-1.5

U.VIGNESH

2303A51964

BATCH-24

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

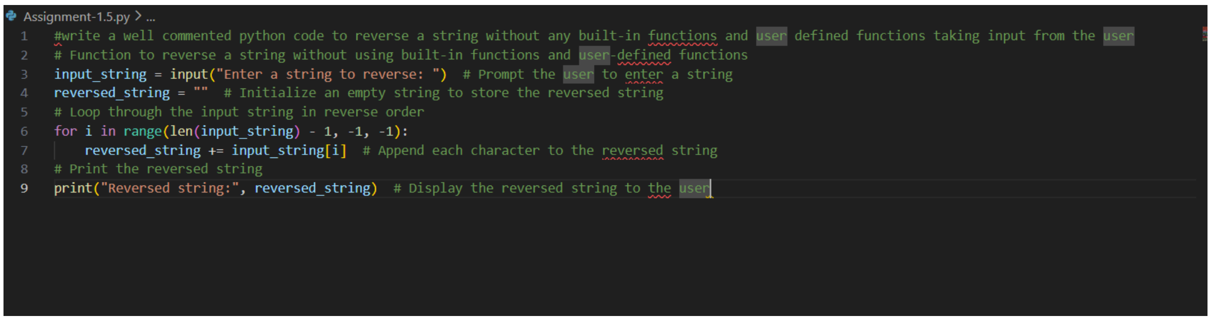
❖ Expected Output

➢ Correct reversed string

➢ Screenshots showing Copilot-generated code suggestions

➢ Sample inputs and outputs

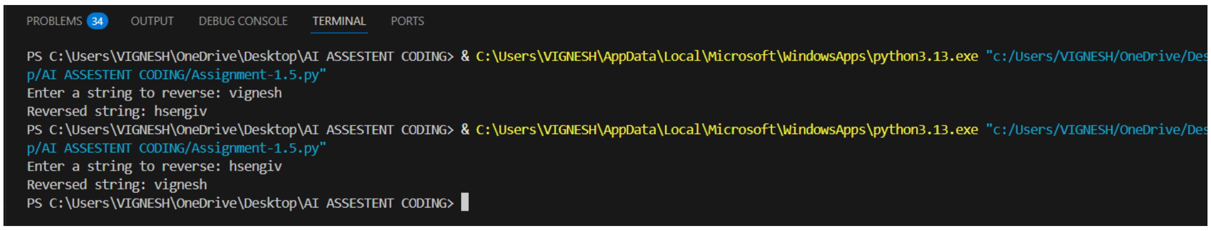
CODE:



Assignment-1.5.py > ...

```python
#write a well commented python code to reverse a string without any built-in functions and user defined functions taking input from the user
# Function to reverse a string without using built-in functions and user-defined functions
input_string = input("Enter a string to reverse: ")  # Prompt the user to enter a string
reversed_string = ""  # Initialize an empty string to store the reversed string
# Loop through the input string in reverse order
for i in range(len(input_string) - 1, -1, -1):
    reversed_string += input_string[i]  # Append each character to the reversed string
# Print the reversed string
print("Reversed string:", reversed_string)  # Display the reversed string to the user
```

OUTPUT:



```
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING> & C:\Users\VIGNESH\AppData\Local\Microsoft\WindowsApps\python3.13.exe "c:/Users/VIGNESH/OneDrive/Des
p/AI ASSESTENT CODING/Assignment-1.5.py"
Enter a string to reverse: vignesh
Reversed string: hsengiv
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING> & C:\Users\VIGNESH\AppData\Local\Microsoft\WindowsApps\python3.13.exe "c:/Users/VIGNESH/OneDrive/Des
p/AI ASSESTENT CODING/Assignment-1.5.py"
Enter a string to reverse: hsengiv
Reversed string: vignesh
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING>
```

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

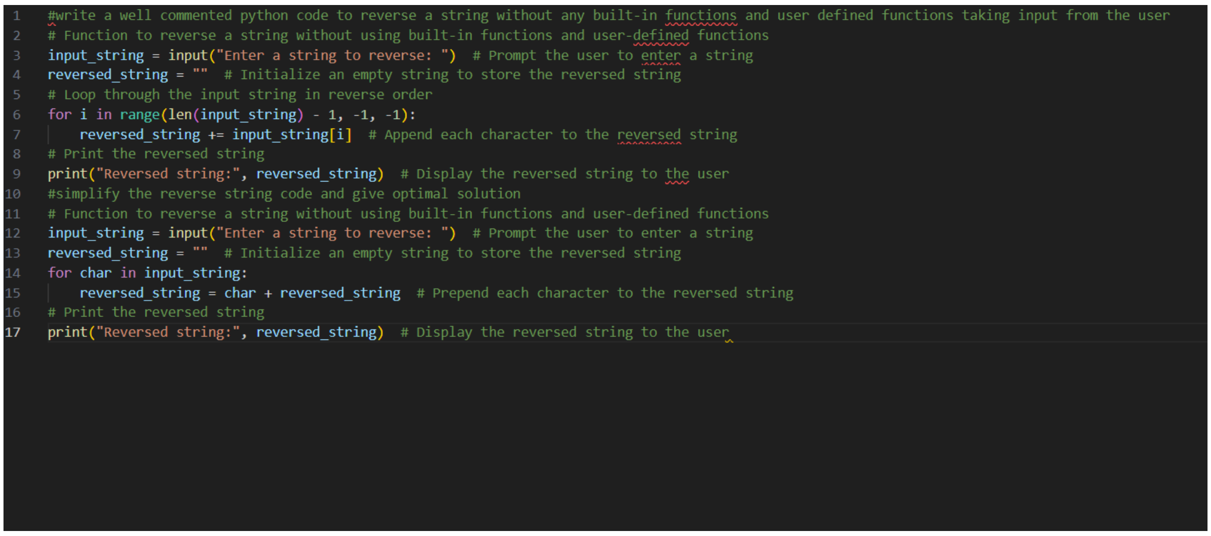
“optimize this code”, “simplify logic”, or “make it more readable”

❖ Expected Output

➢ Original and optimized code versions

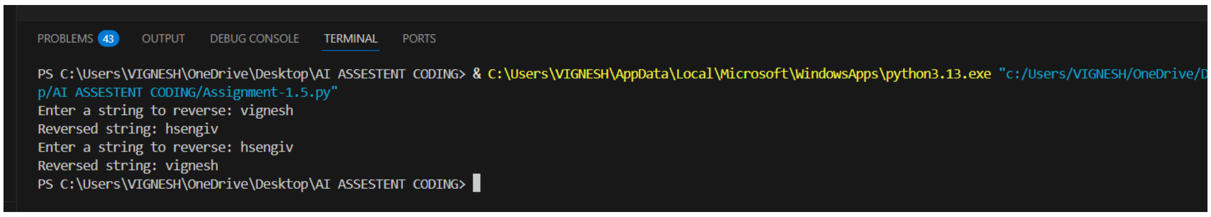
➢ Explanation of how the improvements reduce time complexity

CODE:



```python
#write a well commented python code to reverse a string without any built-in functions and user defined functions taking input from the user
# Function to reverse a string without using built-in functions and user-defined functions
input_string = input("Enter a string to reverse: ")  # Prompt the user to enter a string
reversed_string = ""  # Initialize an empty string to store the reversed string
# Loop through the input string in reverse order
for i in range(len(input_string) - 1, -1, -1):
    reversed_string += input_string[i]  # Append each character to the reversed string
# Print the reversed string
print("Reversed string:", reversed_string)  # Display the reversed string to the user
#simplify the reverse string code and give optimal solution
# Function to reverse a string without using built-in functions and user-defined functions
input_string = input("Enter a string to reverse: ")  # Prompt the user to enter a string
reversed_string = ""  # Initialize an empty string to store the reversed string
for char in input_string:
    reversed_string = char + reversed_string  # Prepend each character to the reversed string
# Print the reversed string
print("Reversed string:", reversed_string)  # Display the reversed string to the user
```

OUTPUT:



```
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING> & C:\Users\VIGNESH\AppData\Local\Microsoft\WindowsApps\python3.13.exe "c:/Users/VIGNESH/OneDrive/D
p/AI ASSESTENT CODING/Assignment-1.5.py"
Enter a string to reverse: vignesh
Reversed string: hsengiv
Enter a string to reverse: hsengiv
Reversed string: vignesh
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT 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)

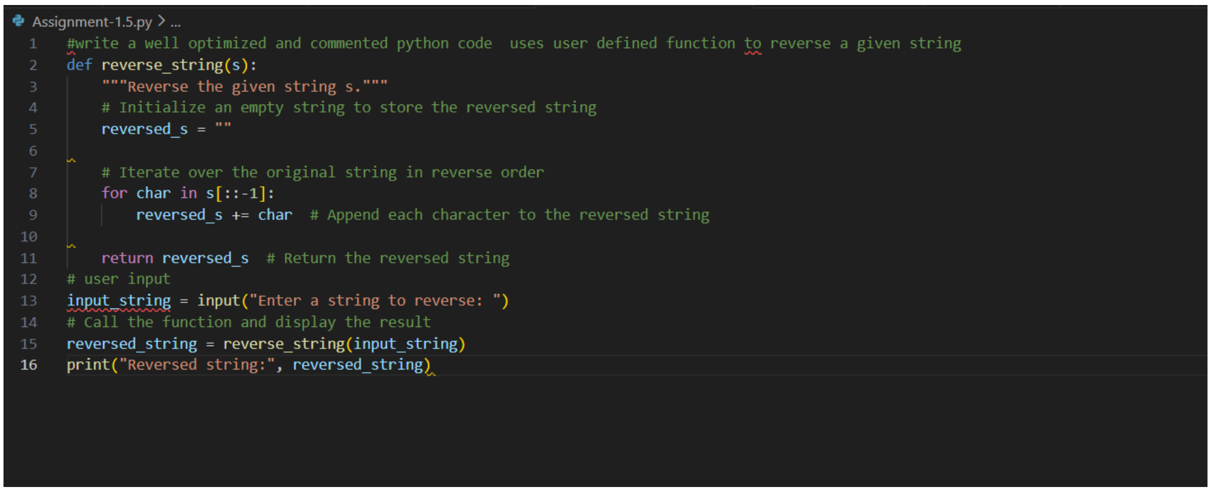
❖ Expected Output

➢ Correct function-based implementation

➢ Screenshots documenting Copilot’s function generation

➢ Sample test cases and outputs

CODE:

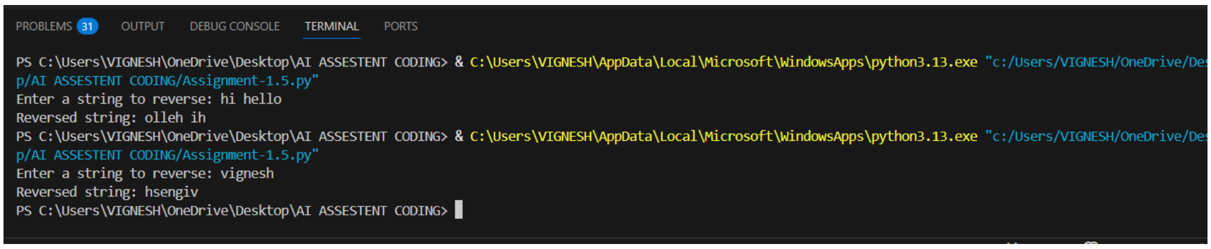


```
Assignment-1.5.py > ...
#write a well optimized and commented python code  uses user defined function to reverse a given string
def reverse_string(s):
    """Reverse the given string s."""
    # Initialize an empty string to store the reversed string
    reversed_s = ""

    # Iterate over the original string in reverse order
    for char in s[::-1]:
        reversed_s += char  # Append each character to the reversed string

    return reversed_s  # Return the reversed string
# user input
input_string = input("Enter a string to reverse: ")
# Call the function and display the result
reversed_string = reverse_string(input_string)
print("Reversed string:", reversed_string)
```

OUTPUT:



```
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING> & C:\Users\VIGNESH\AppData\Local\Microsoft\WindowsApps\python3.13.exe "c:/Users/VIGNESH/OneDrive/Des
p/AI ASSESTENT CODING/Assignment-1.5.py"
Enter a string to reverse: hi hello
Reversed string: olleh ih
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING> & C:\Users\VIGNESH\AppData\Local\Microsoft\WindowsApps\python3.13.exe "c:/Users/VIGNESH/OneDrive/Des
p/AI ASSESTENT CODING/Assignment-1.5.py"
Enter a string to reverse: vignesh
Reversed string: hsengiv
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING>
```

Task 4: Comparative Analysis – Procedural vs Modular Approach (With vs

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

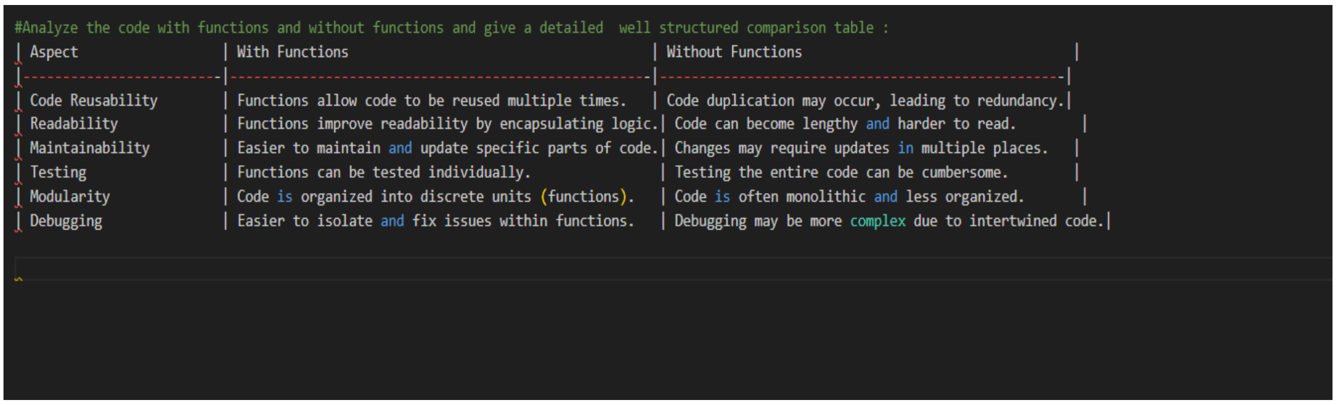
➢ Debugging ease

➢ Suitability for large-scale applications

❖ Expected Output

Comparison table or short analytical report

OUTPUT:



```
#Analyze the code with functions and without functions and give a detailed  well structured comparison table :
| Aspect                  | With Functions                                  | Without Functions                              |
|-------------------------|-------------------------------------------------|------------------------------------------------|
| Code Reusability        | Functions allow code to be reused multiple times.   | Code duplication may occur, leading to redundancy.|
| Readability             | Functions improve readability by encapsulating logic.| Code can become lengthy and harder to read.        |
| Maintainability         | Easier to maintain and update specific parts of code.| Changes may require updates in multiple places.   |
| Testing                 | Functions can be tested individually.               | Testing the entire code can be cumbersome.        |
| Modularity              | Code is organized into discrete units (functions).  | Code is often monolithic and less organized.       |
| Debugging               | Easier to isolate and fix issues within functions.  | Debugging may be more complex due to intertwined code.|
```

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

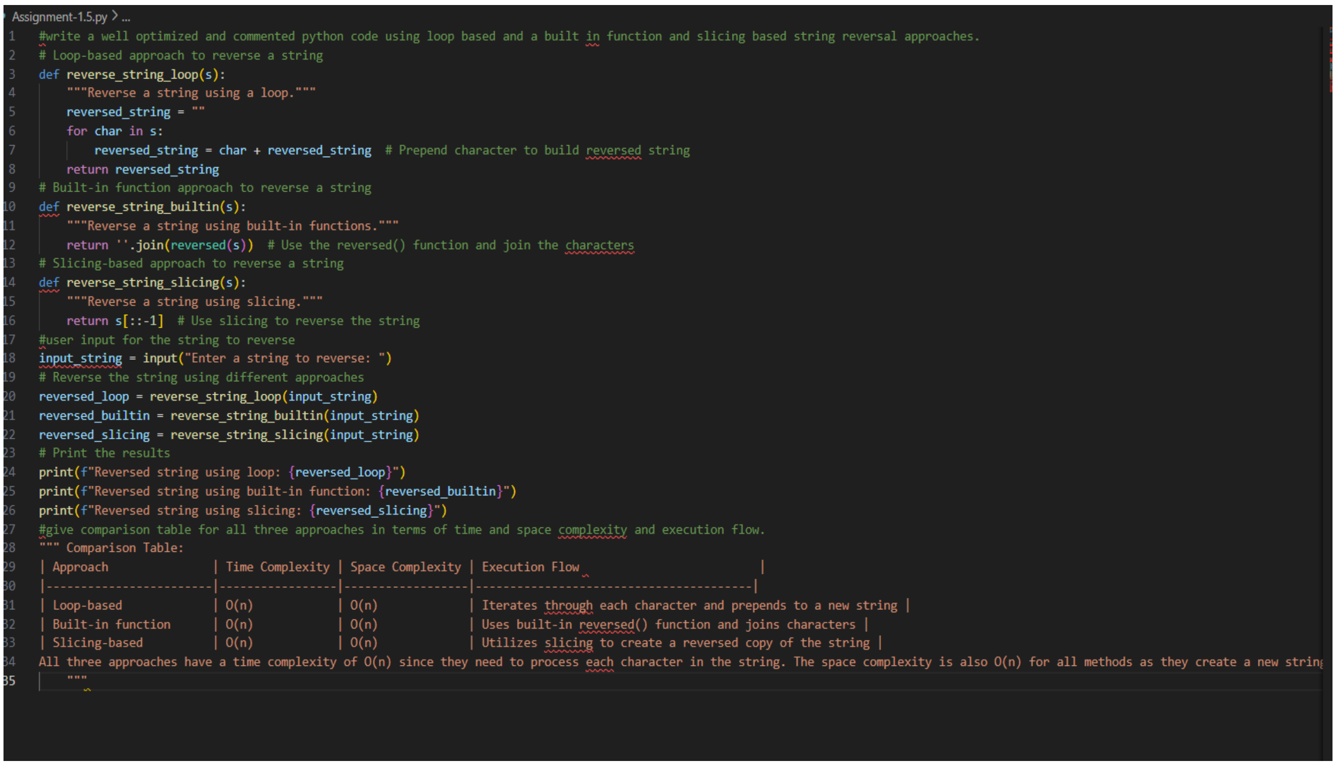
➢ Comparison discussing:

▪ Execution flow

▪ Time complexity

▪ Performance for large inputs

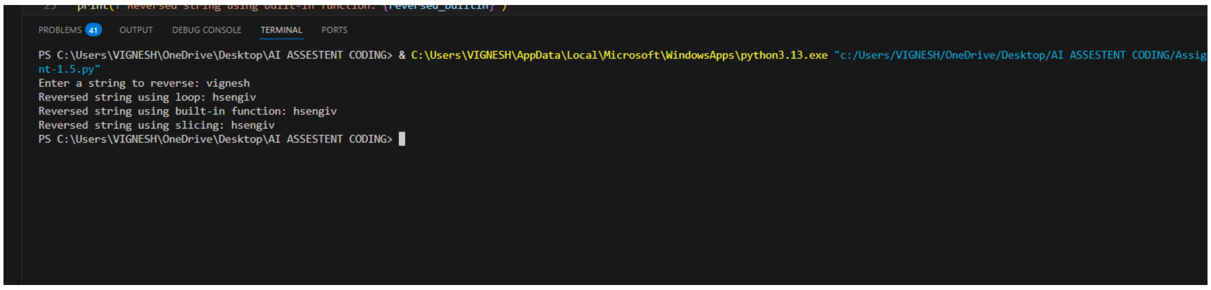
▪ When each approach is appropriate



Assignment-1.5.py > ...

```python
#write a well optimized and commented python code using loop based and a built in function and slicing based string reversal approaches.
# Loop-based approach to reverse a string
def reverse_string_loop(s):
    """Reverse a string using a loop."""
    reversed_string = ""
    for char in s:
        reversed_string = char + reversed_string  # Prepend character to build reversed string
    return reversed_string
# Built-in function approach to reverse a string
def reverse_string_builtin(s):
    """Reverse a string using built-in functions."""
    return ''.join(reversed(s))  # Use the reversed() function and join the characters
# Slicing-based approach to reverse a string
def reverse_string_slicing(s):
    """Reverse a string using slicing."""
    return s[::-1]  # Use slicing to reverse the string
#user input for the string to reverse
input_string = input("Enter a string to reverse: ")
# Reverse the string using different approaches
reversed_loop = reverse_string_loop(input_string)
reversed_builtin = reverse_string_builtin(input_string)
reversed_slicing = reverse_string_slicing(input_string)
# Print the results
print(f"Reversed string using loop: {reversed_loop}")
print(f"Reversed string using built-in function: {reversed_builtin}")
print(f"Reversed string using slicing: {reversed_slicing}")
#give comparison table for all three approaches in terms of time and space complexity and execution flow.
""" Comparison Table:
| Approach                 | Time Complexity | Space Complexity | Execution Flow                           |
|------------------------|-----------------|------------------|-----------------------------------------|
| Loop-based               | O(n)            | O(n)             | Iterates through each character and prepends to a new string |
| Built-in function        | O(n)            | O(n)             | Uses built-in reversed() function and joins characters |
| Slicing-based            | O(n)            | O(n)             | Utilizes slicing to create a reversed copy of the string |
All three approaches have a time complexity of O(n) since they need to process each character in the string. The space complexity is also O(n) for all methods as they create a new strin
    """
```

OUTPUT:



```
    print(f"Reversed string using built-in function: {reversed_builtin}")
```

PROBLEMS 41   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

```
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING> & C:\Users\VIGNESH\AppData\Local\Microsoft\WindowsApps\python3.13.exe "c:/Users/VIGNESH/OneDrive/Desktop/AI ASSESTENT CODING/Assig
nt-1.5.py"
Enter a string to reverse: vignesh
Reversed string using loop: hsengiv
Reversed string using built-in function: hsengiv
Reversed string using slicing: hsengiv
PS C:\Users\VIGNESH\OneDrive\Desktop\AI ASSESTENT CODING> 
```