ASSIGNMENT 1.3

Name: B.Bhargava Chary

HT NO: 2303A51747

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

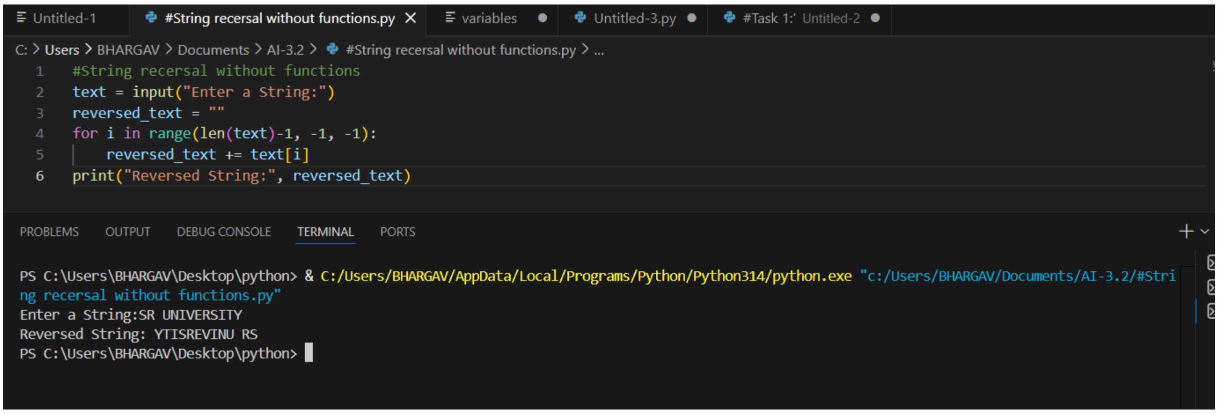
➢ Does not use any user-defined functions

❖ Expected Output

➢ Correct reversed string

➢ Screenshots showing Copilot-generated code suggestions

➢ Sample inputs and outputs



Untitled-1
#String recersal without functions.py
variables
Untitled-3.py
#Task 1:' Untitled-2
C: > Users > BHARGAV > Documents > AI-3.2 > #String recersal without functions.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```python
#String recersal without functions
text = input("Enter a String:")
reversed_text = ""
for i in range(len(text)-1, -1, -1):
    reversed_text += text[i]
print("Reversed String:", reversed_text)
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Stri
ng recersal without functions.py"
Enter a String:SR UNIVERSITY
Reversed String: YTISREVINU RS
PS C:\Users\BHARGAV\Desktop\python>
```

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:

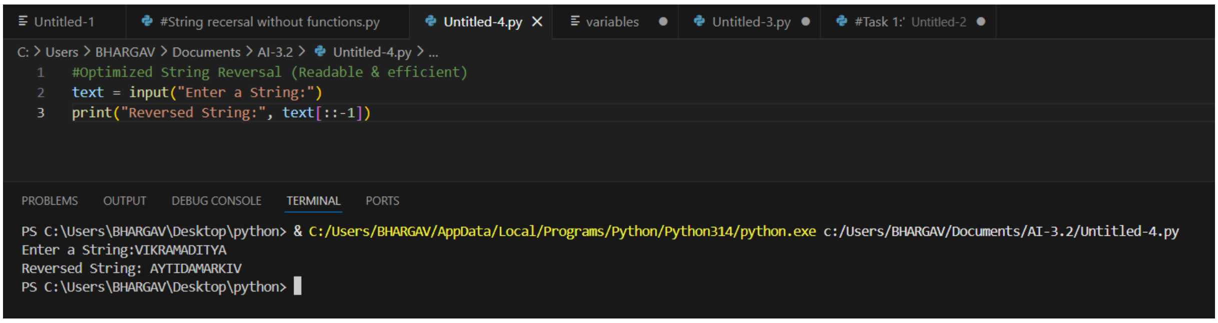
Prompt Copilot with phrases like

“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



```python
#Optimized String Reversal (Readable & efficient)
text = input("Enter a String:")
print("Reversed String:", text[::-1])
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/Untitled-4.py
Enter a String:VIKRAMADITYA
Reversed String: AYTIDAMARKIV
PS C:\Users\BHARGAV\Desktop\python> 
```

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

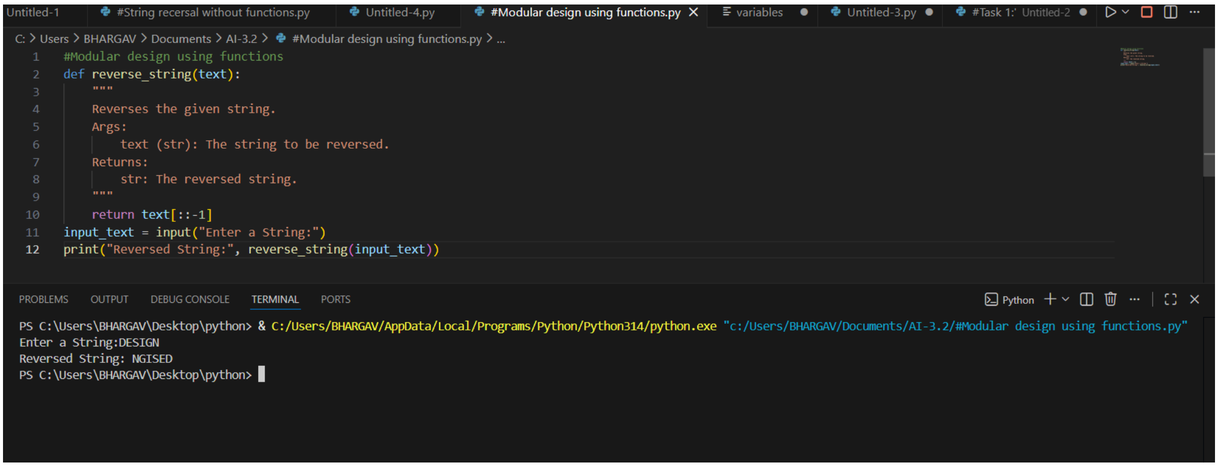
➢ Includes meaningful comments (AI-assisted)

❖ Expected Output

➢ Correct function-based implementation

➢ Screenshots documenting Copilot’s function generation

➢ Sample test cases and outputs


Untitled-1
#String recersal without functions.py
Untitled-4.py
#Modular design using functions.py
variables
Untitled-3.py
#Task 1:' Untitled-2
C: > Users > BHARGAV > Documents > AI-3.2 > #Modular design using functions.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Python

```python
#Modular design using functions
def reverse_string(text):
    """
    Reverses the given string.
    Args:
        text (str): The string to be reversed.
    Returns:
        str: The reversed string.
    """
    return text[::-1]
input_text = input("Enter a String:")
print("Reversed String:", reverse_string(input_text))
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Modular design using functions.py"
Enter a String:DESIGN
Reversed String: NGISED
PS C:\Users\BHARGAV\Desktop\python> 
```

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

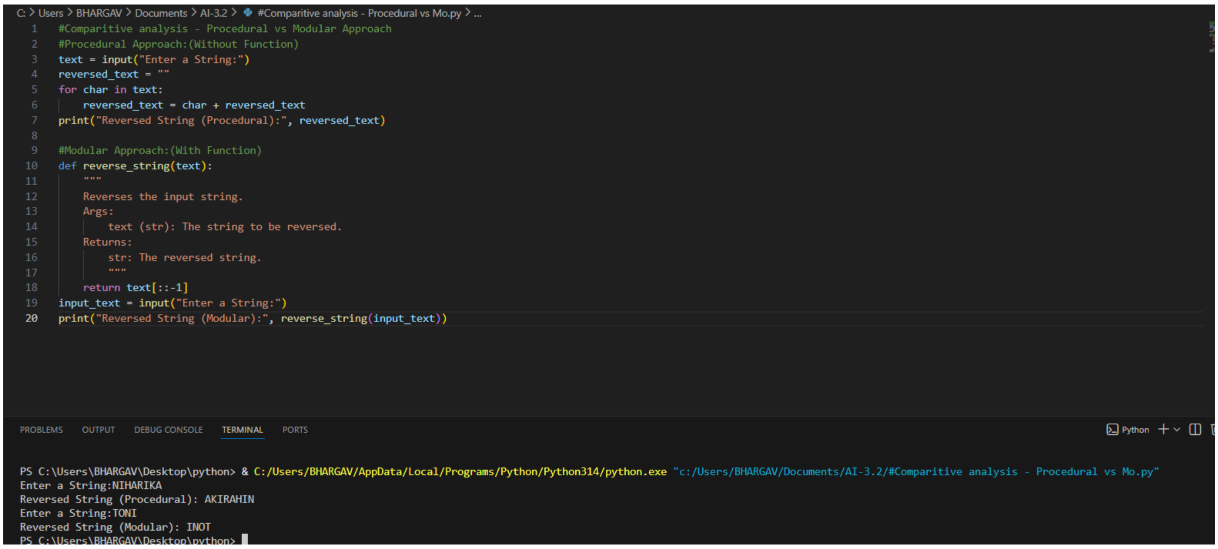
➢ Reusability

➢ Debugging ease

➢ Suitability for large-scale applications

❖ Expected Output

Comparison table or short analytical report


C: > Users > BHARGAV > Documents > AI-3.2 > #Comparitive analysis - Procedural vs Mo.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Python

```python
#Comparitive analysis - Procedural vs Modular Approach
#Procedural Approach:(Without Function)
text = input("Enter a String:")
reversed_text = ""
for char in text:
    reversed_text = char + reversed_text
print("Reversed String (Procedural):", reversed_text)

#Modular Approach:(With Function)
def reverse_string(text):
    """
    Reverses the input string.
    Args:
        text (str): The string to be reversed.
    Returns:
        str: The reversed string.
        """
    return text[::-1]
input_text = input("Enter a String:")
print("Reversed String (Modular):", reverse_string(input_text))
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Comparitive analysis - Procedural vs Mo.py"
Enter a String:NIHARIKA
Reversed String (Procedural): AKIRAHIN
Enter a String:TONI
Reversed String (Modular): INOT
PS C:\Users\BHARGAV\Desktop\python> 
```

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

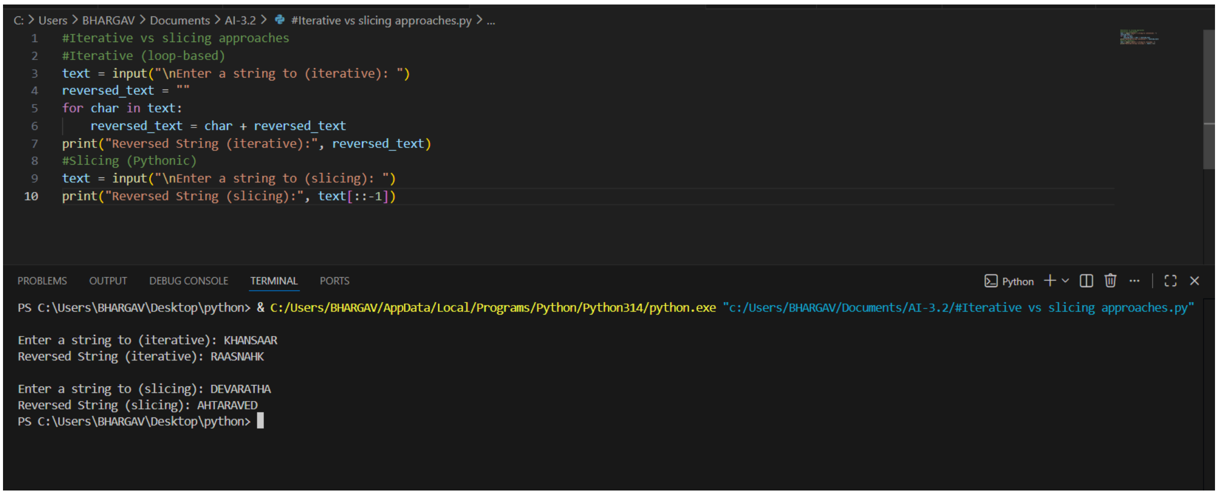
➢ Comparison discussing:

▪ Execution flow

▪ Time complexity

▪ Performance for large inputs

▪ When each approach is appropriate



C: > Users > BHARGAV > Documents > AI-3.2 > #Iterative vs slicing approaches.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Python

```python
#Iterative vs slicing approaches
#Iterative (loop-based)
text = input("\nEnter a string to (iterative): ")
reversed_text = ""
for char in text:
    reversed_text = char + reversed_text
print("Reversed String (iterative):", reversed_text)
#Slicing (Pythonic)
text = input("\nEnter a string to (slicing): ")
print("Reversed String (slicing):", text[::-1])
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Iterative vs slicing approaches.py"

Enter a string to (iterative): KHANSAAR
Reversed String (iterative): RAASNAHK

Enter a string to (slicing): DEVARATHA
Reversed String (slicing): AHTARAVED
PS C:\Users\BHARGAV\Desktop\python> 
```