AI-Assisted String Reversal Using GitHub Copilot

Name: Sushanth

Batch : 11

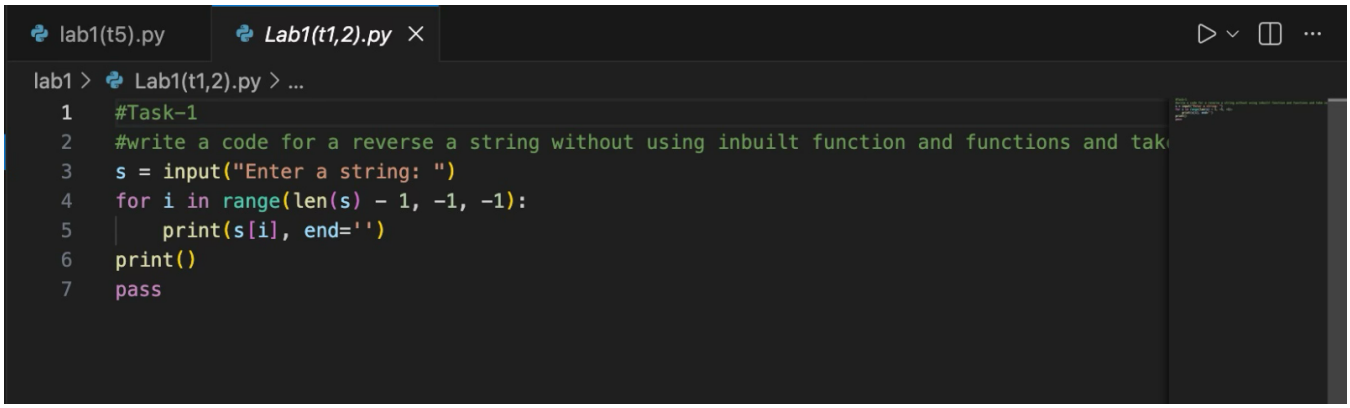
Task 1 & 2: AI-Generated Logic Without Modularization

Scenario

A basic text-processing utility is developed for a messaging application.

Copilot Prompt Used

```
# write a code to reverse a string without using any functions
# accept user input for the string to be reversed
```



Sample Output

```
/usr/bin/python3 "/Users/sushanth/Desktop/Ai Coding/lab1/Lab1(t1,2).py"
sushanth@Sushanths-MacBook-Air / % /usr/bin/python3 "/Users/sushanth/Desktop/Ai Coding/lab1/Lab1(t1,2).py"
Enter a string: SushanthVirat
tariVhtnahsuS
```

Explanation

The program reads user input and reverses the string using a loop without defining any functions. The logic is implemented directly in the main code.

Task 3: Modular Design Using Functions

Scenario

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

Copilot Prompt Used

```
# write code with function to reverse a string and comment each
line and function call and no sample input and output
```

Code

```python
#Task-3
# Algorithm: iterate over the string indices from last to first and build the reversed st
# This keeps memory usage low by appending characters directly to the result string.
# Note: s[::-1] or ''.join(reversed(s)) are simpler and typically faster alternatives.
#use a user defined function to reverse a string and return the reversed string.
def reverse_string(s):
    rev = ""
    for i in range(len(s)-1, -1, -1):
        rev += s[i]
    return rev

s = input("Enter a string: ")
rev = reverse_string(s)
print("Reversed string:", rev)
```

Sample Output

```
sushanth@Sushanths-MacBook-Air / % /usr/bin/python3 "/Users/sushanth/Desktop/Ai Coding/lab1/lab1(t3).py"
Enter a string: SushanthSandesh
Reversed string: hsednaShtnahsuS
sushanth@Sushanths-MacBook-Air / %
```

Explanation

Using a function improves reusability, readability, and maintainability. The logic can now be reused across different modules.

Task 4: Comparative Analysis – With vs Without Functions

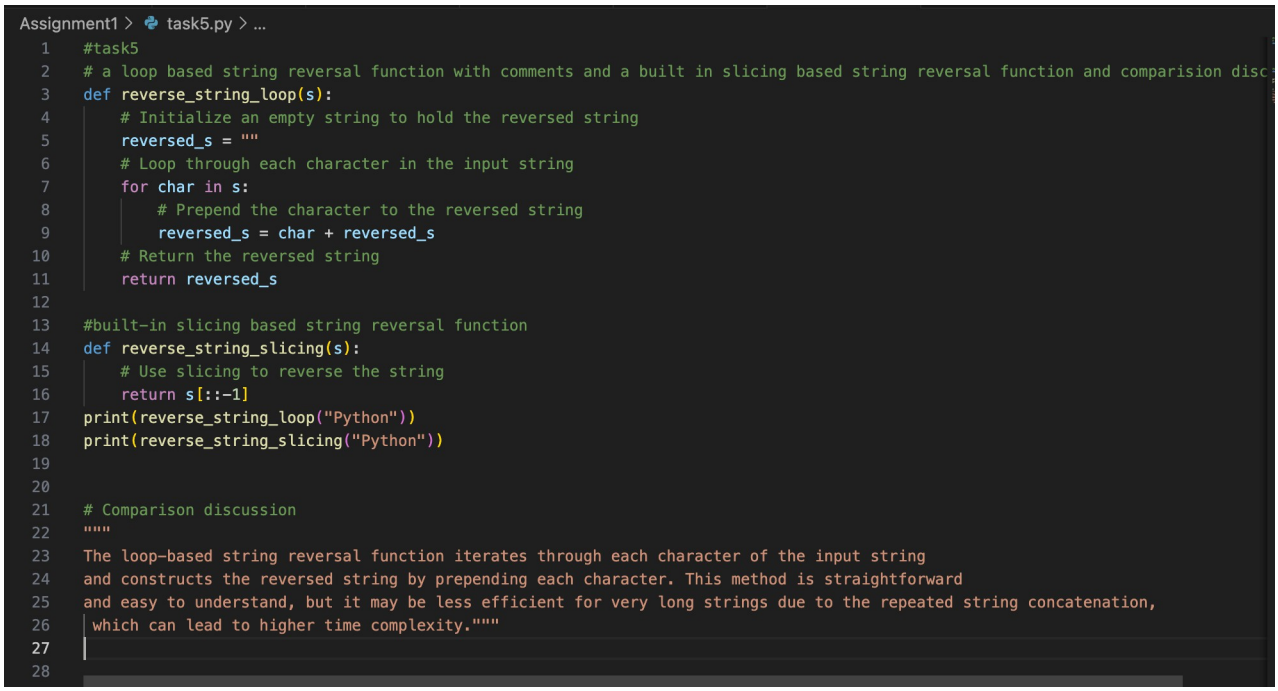
Aspect	Without Functions	With Functions
Code Clarity	Less organized	More structured
Reusability	Cannot be reused	Easily reusable
Debugging	Harder	Easier
Scalability	Not suitable	Suitable
Best Use	Small scripts	Large applications

Conclusion

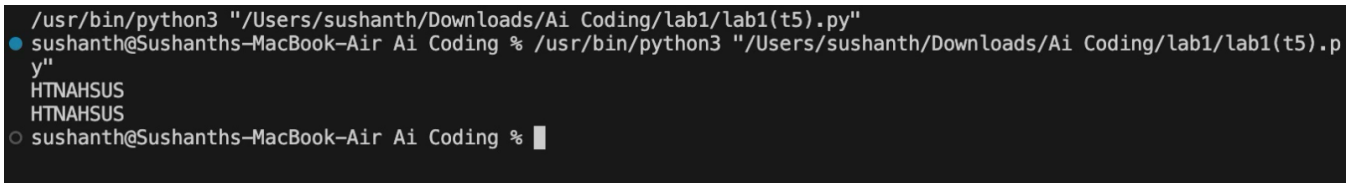
Function-based design is preferred for large-scale applications due to better organization and reusability.

Task 5: Loop-Based vs Built-In String Reversal

Copilot Prompt Use



Output:



Comparison Discussion

The loop-based approach reverses the string by checking each character one by one. It helps beginners understand how string reversal works.

The slicing-based approach uses Python's built-in feature to reverse the string. It is shorter and easier to read.

Both methods take the same amount of time, which is **O(n)**, where *n* is the length of the string.

However, slicing works faster for large strings because it is optimized inside Python.

The loop-based method is good for learning, while the slicing method is better for real applications.

Conclusion

GitHub Copilot helped in writing, improving, and comparing different string reversal programs. Using functions and built-in features makes the code easier to read, faster, and better for large programs.

Declaration

All the code and explanations were created using GitHub Copilot and were checked manually to ensure correctness.