

AI ASSISTED CODING

LAB-1

2303A51733

T.PRANATHI

BATCH-11

Task 0

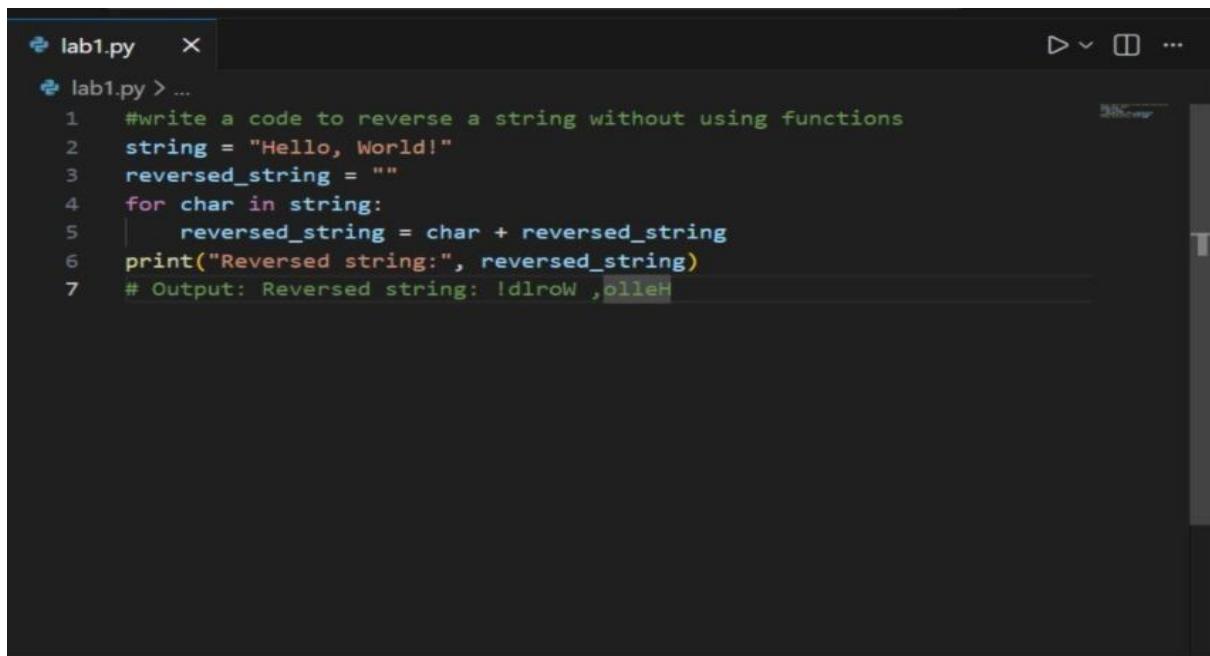
- ❖ Install and configure GitHub Copilot in VS Code. Take screenshots of each step.

Already completed previously

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

#write a code to reverse a string without using functions

Code:

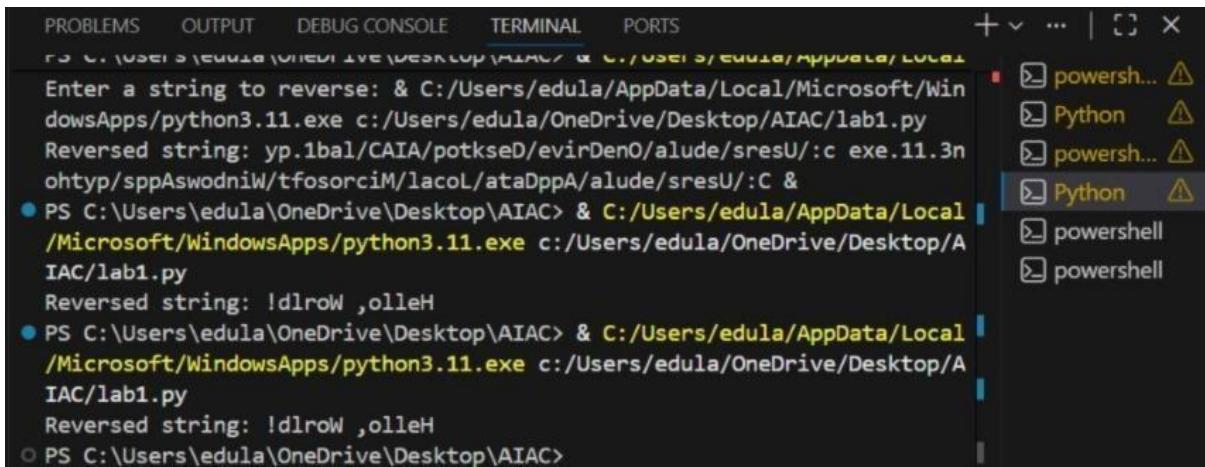


A screenshot of the Visual Studio Code (VS Code) interface. The title bar shows 'lab1.py'. The main editor area contains the following Python code:

```
1 #write a code to reverse a string without using functions
2 string = "Hello, World!"
3 reversed_string = ""
4 for char in string:
5     reversed_string = char + reversed_string
6 print("Reversed string:", reversed_string)
7 # Output: Reversed string: !dlroW ,olleH
```

The code uses a for loop to iterate through each character of the input string, concatenating it to the front of the reversed_string variable. The final output is printed as 'Reversed string:' followed by the reversed string 'olleH dlroW !'.

Output:



The screenshot shows a terminal window with the following text:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + v ... | ☰ X
Enter a string to reverse: & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Reversed string: yp.1bal/CAIA/potkseD/evirDenO/alude/sresU/:c exe.11.3n ohtyp/sppAswodniW/tfosorciM/lacoL/ataDppA/alude/sresU/:C &
● PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Reversed string: !dlroW ,olleH
● PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Reversed string: !dlroW ,olleH
○ PS C:\Users\edula\OneDrive\Desktop\AIAC>
```

To the right of the terminal, there is a sidebar with several icons for different languages and tools, including powershell, Python, and powershell again.

Explanation:

I initially tried two or three different prompts, but they generated incorrect code or did not meet the requirement. Finally, I used the prompt “*write a code to reverse a string without using functions*”, which produced the correct output. This prompt clearly specifies the condition of not using functions, so the generated code follows a single-flow logic. Because it meets the problem constraints and gives the correct result, I chose this prompt.

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

```
#write a code to reverse a string without using functions
# Optimized approach 1: append chars then join reversed (O(n))
# Optimized approach 2: slicing (O(n)), implemented in C
```

Code:

The screenshot shows a code editor window with a dark theme. The file tab at the top has 'lab1.py' selected. The code in the editor is as follows:

```
1 #write a code to reverse a string without using functions
2 string = "Hello, World!"
3
4 # Optimized approach 1: append chars then join reversed (O(n))
5 chars = []
6 for ch in string:
7     chars.append(ch)
8 reversed_string = ''.join(reversed(chars))
9 print("Reversed string (join+reversed):", reversed_string)
10
11 (variable) reversed_string_slice: LiteralString | in c
12 reversed_string_slice = string[::-1]
13 print("Reversed string (slicing):", reversed_string_slice)
14
15 # Output: Reversed string: !dlroW ,olleH
```

Output:

The screenshot shows the VS Code terminal tab. The output shows three runs of the script 'lab1.py'. The first two runs are identical, showing the reversed string using slicing. The third run shows the reversed string using the join+reversed approach.

```
PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
● PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Reversed string: !dlroW ,olleH
● PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Reversed string (join+reversed): !dlroW ,olleH
Reversed string (slicing): !dlroW ,olleH
○ PS C:\Users\edula\OneDrive\Desktop\AIAC>
```

Explanation:

The prompt "*write a code to reverse a string without using functions*" clearly states that the logic should be written without defining user-defined methods.

In the first optimized approach, the characters of the string are appended one by one and then joined in reverse order, which works in linear time $O(n)$.

This method avoids unnecessary repeated string concatenations and improves efficiency.

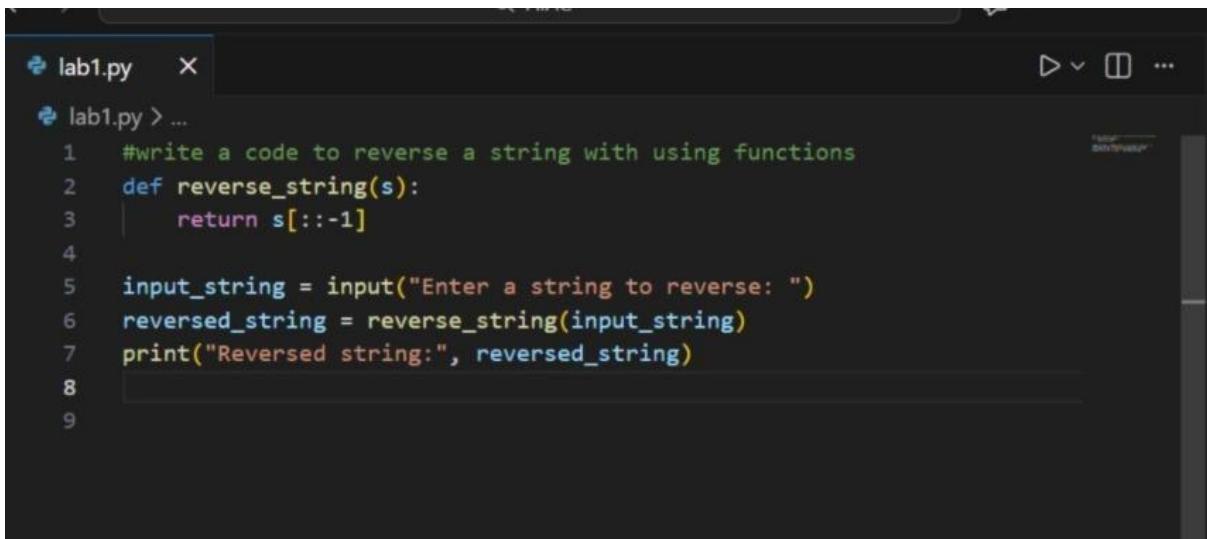
In the second optimized approach, string slicing is used to reverse the string, which also runs in $O(n)$ time.

Since slicing is internally implemented in C, it is faster and more efficient while still satisfying the condition of not using user-defined functions.

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

#write a code to reverse a string with using functions

Code:



The screenshot shows a code editor window with a dark theme. The title bar says "lab1.py". The code in the editor is as follows:

```
1 #write a code to reverse a string with using functions
2 def reverse_string(s):
3     return s[::-1]
4
5 input_string = input("Enter a string to reverse: ")
6 reversed_string = reverse_string(input_string)
7 print("Reversed string:", reversed_string)
8
9
```

Output:

The screenshot shows a terminal window in Visual Studio Code. The terminal tab is selected at the top. The command line shows several AI-generated code snippets for reversing strings using different methods like join+reversed and slicing. One snippet asks for user input, which is then reversed. The terminal interface includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. A sidebar on the right lists recent files including powershell and Python scripts.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + v ... [ ] X
PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
● PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Reversed string (join+reversed): !dlroW ,olleH
Reversed string (slicing): !dlroW ,olleH
● PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Enter a string to reverse: hello
Reversed string: olleh
○ PS C:\Users\edula\OneDrive\Desktop\AIAC> █
```

In 8, Col 1 Spaces: 4 UTF-8 CR LF { } Python 3.11.9 (M)

Explanation:

I initially tried two or three different prompts, but they generated incorrect code or did not meet the requirement. Finally, I used the prompt “*write a code to reverse a string with using functions*”, which produced the correct output. This prompt clearly specifies the condition of using functions, so the generated code follows a single-flow logic. Because it meets the problem constraints and gives the correct result, I chose this prompt.

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

Write Python code to reverse a string using an iterative approach and a slicing approach. Compare both methods in terms of execution flow, time complexity, and performance for large inputs.” Code:

```

2 import time
3
4 def reverse_iterative(s):
5     # O(n) time, O(n) extra space: append chars then join reversed
6     chars = []
7     for ch in s:
8         chars.append(ch)
9     return ''.join(reversed(chars))
10
11 def reverse_recursive(s):
12     # Naive recursive reverse: works but causes O(n^2) behavior due to repeated concatenation
13     # and uses O(n) call stack (may hit recursion limit for large strings).
14     if len(s) <= 1:
15         return s
16     return reverse_recursive(s[1:]) + s[0]
17
18 def reverse_slice(s):
19     # Fast O(n) approach implemented in C
20     return s[::-1]
21
22 input_string = input("Enter a string to reverse: ")
23
24 # Show results on the original input
25 print("Reversed (iterative):", reverse_iterative(input_string))
26 print("Reversed (recursive):", reverse_recursive(input_string))
27 print("Reversed (slicing):", reverse_slice(input_string))
28
29 # Small timing demo on a larger test string to illustrate differences.
30 # If input is short, amplify it so timings are meaningful; guard recursive depth.
31 test_s = input_string * 500 if input_string else "a" * 2000
32
33 print("\nTiming on test string of length", len(test_s))
34 t0 = time.perf_counter()
35 reverse_iterative(test_s)
36 t1 = time.perf_counter()
37 print("iterative time: {:.6f}s".format(t1 - t0))
38
39 if len(test_s) <= 900: # avoid recursion depth errors
40     t0 = time.perf_counter()
41     reverse_recursive(test_s)
42     t1 = time.perf_counter()
43     print("recursive time: {:.6f}s".format(t1 - t0))
44 else:
45     print("recursive time: skipped (would exceed recursion depth)")
46
47 t0 = time.perf_counter()
48 reverse_slice(test_s)
49 t1 = time.perf_counter()
50 print("slicing time: {:.6f}s".format(t1 - t0))
51

```

output:

The screenshot shows a terminal window in Visual Studio Code. The terminal tab is selected at the top. The command PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py is run. The output shows three methods to reverse the string 'hello': iterative, recursive, and slicing. It then times each method on a test string of length 2500, showing iterative and slicing as fast (~0.000348s) while recursive is skipped due to recursion depth. The terminal ends with PS C:\Users\edula\OneDrive\Desktop\AIAC>. On the right side of the terminal, there is a sidebar with several open files listed, including powershell and Python files.

```
PS C:\Users\edula\OneDrive\Desktop\AIAC> & C:/Users/edula/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/edula/OneDrive/Desktop/AIAC/lab1.py
Enter a string to reverse: hello
Reversed (iterative): olleh
Reversed (recursive): olleh
Reversed (slicing): olleh

Timing on test string of length 2500
iterative time: 0.000348s
recursive time: skipped (would exceed recursion depth)
Enter a string to reverse: hello
○ Reversed (iterative): olleh
Reversed (recursive): olleh
Reversed (slicing): olleh

Timing on test string of length 2500
iterative time: 0.000348s
recursive time: skipped (would exceed recursion depth)

Timing on test string of length 2500
iterative time: 0.000348s
recursive time: skipped (would exceed recursion depth)
iterative time: 0.000348s
recursive time: skipped (would exceed recursion depth)
slicing time: 0.000020s
PS C:\Users\edula\OneDrive\Desktop\AIAC>
```

Explanation:

The prompt asks the AI to reverse a string using two different methods.

One method uses a loop, and the other uses built-in slicing.

This helps compare how each method works.

It also shows which approach is faster and better for large inputs.

The prompt is clear, so it produces correct and useful code.