

AI ASSISTANT CODING

Lab Assignment 1.5

Name: A.Naga Koushik

2403A51L22

Batch : 51

Task 1: AI-Generated Logic Without Modularization (Reverse String)

Prompt Used: “write a simple python program on Reverse String without using functions”

The screenshot shows a code editor interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Toolbar:** RUN AND DEBUG, RUN, DEBUG, CHAT.
- Code Editor:** A file named "day 3.py" is open, containing the following Python code:

```
Ai lab > #write a program on reverse string without function
1  String = input("Enter a string: ")
2  reversed_string = ""
3  for char in string:
4      |  reversed_string = char + reversed_string
5  print("Reversed string:", reversed_string)
```
- Terminal:** Shows the command "python day 3.py" and the output "Enter a string: Koushik Reversed string: kihsuok".
- Breakpoints:** A list of breakpoints: Raised Excepti..., Uncaught Except..., User Uncaught... (with the last one checked).
- Bottom Status Bar:** Ln 6, Col 43, Spaces: 4, UTF-8, CRLF, Python, 3.14.2.

- Keeps the program simple
- Suitable for small scripts
- Easy for basic understanding
- No function call overhead

Task 2: AI Code Optimization & Cleanup Original Code:

A screenshot of the Visual Studio Code interface. The top bar shows 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', 'Terminal', 'Help'. The title bar says 'Python training'. The main editor window contains the following Python code:

```
1 #write a program on reverse string without function
2 string = input("Enter a string: ")
3 reversed_string = ""
4 for char in string:
5     reversed_string = char + reversed_string
6 print("Reversed string:", reversed_string)
```

The 'RUN' section in the sidebar has 'Run and Debug' selected. The terminal below shows the output of running the code with the input 'Koushik':

```
Enter a string: Koushik
Reversed string: khsuok
PS C:\Users\girug\Downloads\Python training>
```

The bottom right corner of the interface features an AI agent icon with the text 'Ask about your code'.

Prompt Used: "optimize this code & simplify logic and improve readability"

A screenshot of the Visual Studio Code interface, identical to the first one but with optimized code. The main editor window now contains:

```
1 #write a reverse string program without using function and optyimised code
2 string = input("Enter a string: ")
3 reversed_string = ''
4 for char in string:
5     reversed_string = char + reversed_string
6 print("Reversed string:", reversed_string)
```

The terminal output remains the same as the first screenshot.

Code is cleaner and easier to maintain

The optimized version improves clarity, maintainability, and readability without affecting performance.

Task 3: Modular Design Using AI Assistance (Reverse String with Functions)

Prompt Used: “ Write a simple python program of using with function”

```
#write a program on reverse string with functions with comments
def reverse_string(s):
    if len(s) <= 1:
        return s
    return s[-1] + reverse_string(s[:-1])
input_string = input("Enter a string to reverse: ")
reversed_string = reverse_string(input_string)
print(f"Reversed string: {reversed_string}")
```

Using functions improves reusability because the same logic can be called multiple times.

It also improves readability and debugging.

Modular code is easier to maintain in large projects.

Task 4: With and Without Using Functions(Reverse String)

Prompt Used: “ Write a simple python program of Reverse String using with function and without using function”

The screenshot shows the Visual Studio Code interface with two code files open in tabs: "day 2" and "day 3.py".

day 3.py (Without Function):

```
1 #write a program on reverse string without and with function
2 # Without function
3 string = input("Enter a string: ")
4 reversed_string = ""
5 for char in string:
6     reversed_string = char + reversed_string
7 print("Reversed string without function:", reversed_string)
```

day 3.py (With Function):

```
1 # With function
2 def reverse_string(s):
3     reversed_s = ""
4     for char in s:
5         reversed_s = char + reversed_s
6     return reversed_s
7 input_string = input("Enter a string: ")
8 reversed_str = reverse_string(input_string)
9 print("Reversed string with function:", reversed_str)
```

The terminal below shows the output of running both scripts:

```
Enter a string: Hemanth
Reversed string without function: htnameH
Enter a string: Sandeep
Reversed string with function: pednas
PS C:\Users\girug\Downloads\Python training>
```

The sidebar on the left shows the "RUN AND DEBUG" section with "Run and Debug" selected. The bottom right corner has an AI lab feature with a message: "Ask about your code".

Without using functions: Helps beginners clearly understand the basic logic and step-by-step execution of an Armstrong number program.

Using functions: Makes the code modular, reusable, and easier to read and maintain.

Overall: Using functions follows good programming practices, especially for larger or real-world programs.

Task 5: Iterative vs Recursive AI Code

Prompt Used: “Generate iterative and recursive Reverse String in Python”

The screenshot shows the Visual Studio Code interface. On the left, the sidebar has 'RUN AND DEBUG' and 'RUN' sections. A blue button 'Run and Debug' is highlighted. Below it, there's a note about 'Show automatic Python configurations'. The main editor area contains Python code for reversing a string iteratively:

```
1 #write a reverse string program using recursion
2 def reverse_string(s):
3     if len(s) == 0:
4         return s
5     else:
6         return s[-1] + reverse_string(s[:-1])
7 string = input("Enter a string: ")
8 reversed_str = reverse_string(string)
9 print("Reversed string:", reversed_str)
```

The terminal at the bottom shows the output of running the code:

```
Enter a string: ishuu
Reversed string: uuhsi
PS C:\Users\girug\Downloads\Python training>
```

The status bar at the bottom right shows: Ln 9, Col 40 | Spaces: 4 | UTF-8 | CRLF | { } Python | 3.14.2 |

Execution Flow Explanation

- Iterative version uses loops
- Recursive version uses function calls
- Recursive calls stack memory