AI ASSISTANT CODING

Lab Assignment 1.5

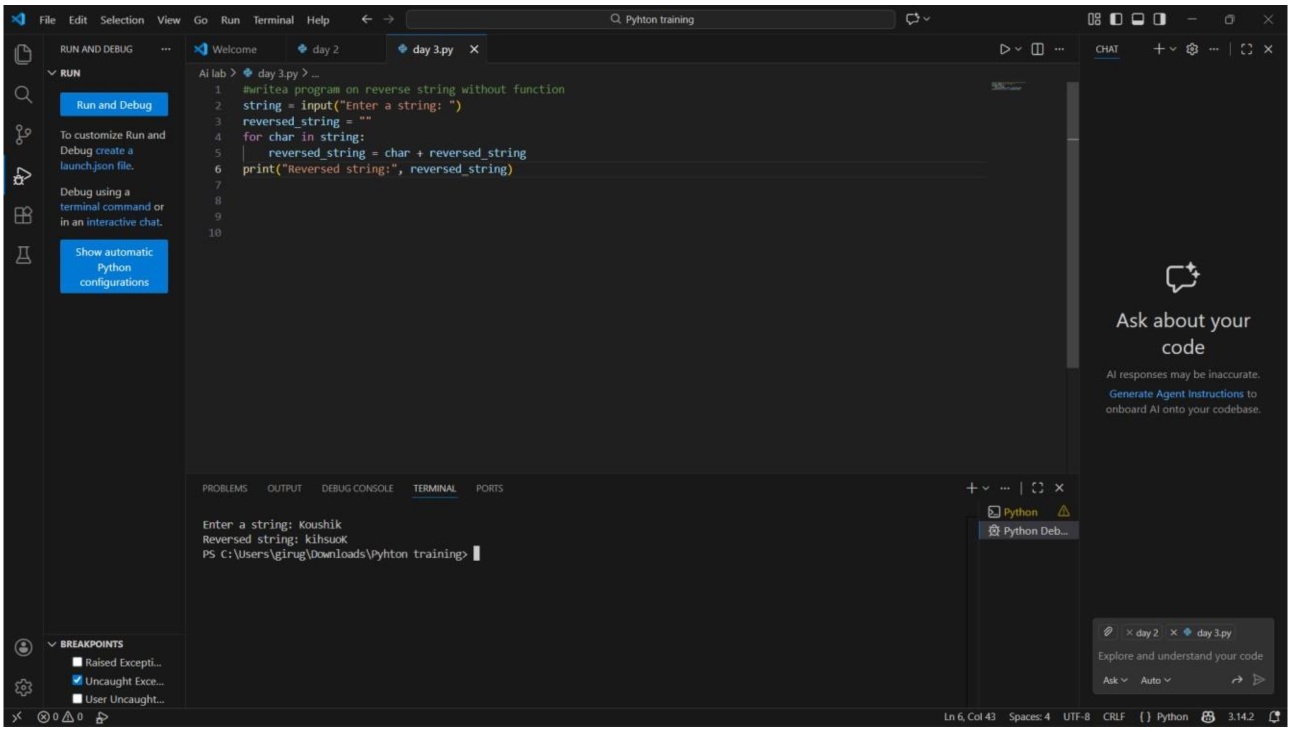
Name: A.Naga Koushik

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"

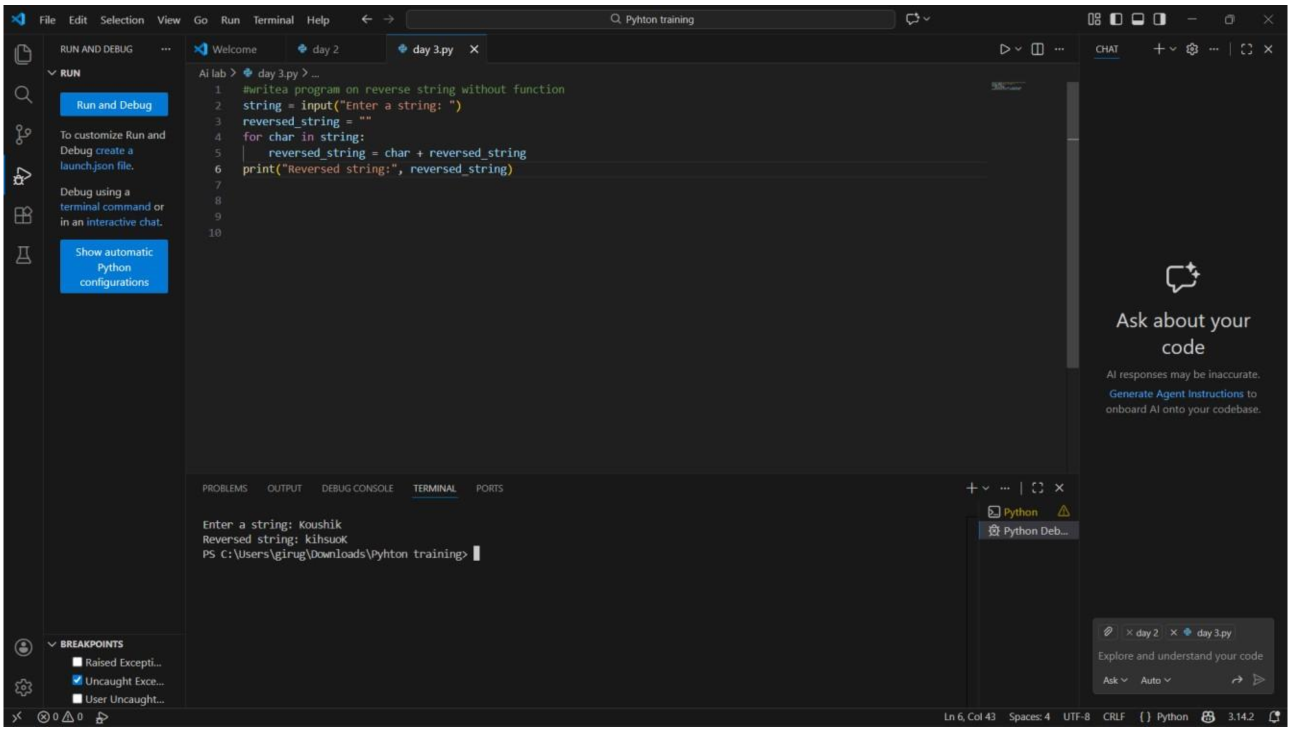


```python
#writea program on reverse string without function
string = input("Enter a string: ")
reversed_string = ""
for char in string:
    reversed_string = char + reversed_string
print("Reversed string:", reversed_string)
```

```
Enter a string: Koushik
Reversed string: kihsuoK
PS C:\Users\girug\Downloads\Pyhton training>
```

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

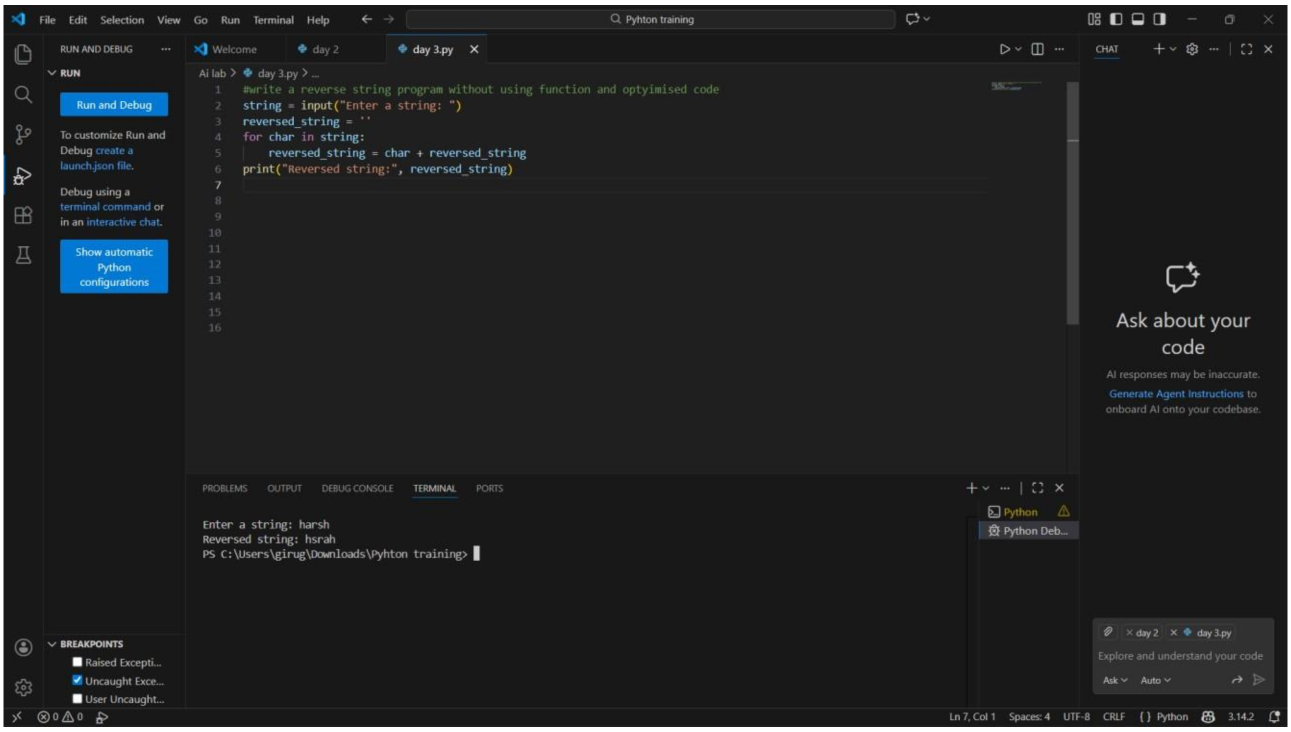
Task 2: AI Code Optimization & Cleanup Original Code:



```python
#writea program on reverse string without function
string = input("Enter a string: ")
reversed_string = ""
for char in string:
    reversed_string = char + reversed_string
print("Reversed string:", reversed_string)
```

```
Enter a string: Koushik
Reversed string: kihsuoK
PS C:\Users\girug\Downloads\Pyhton training>
```

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



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

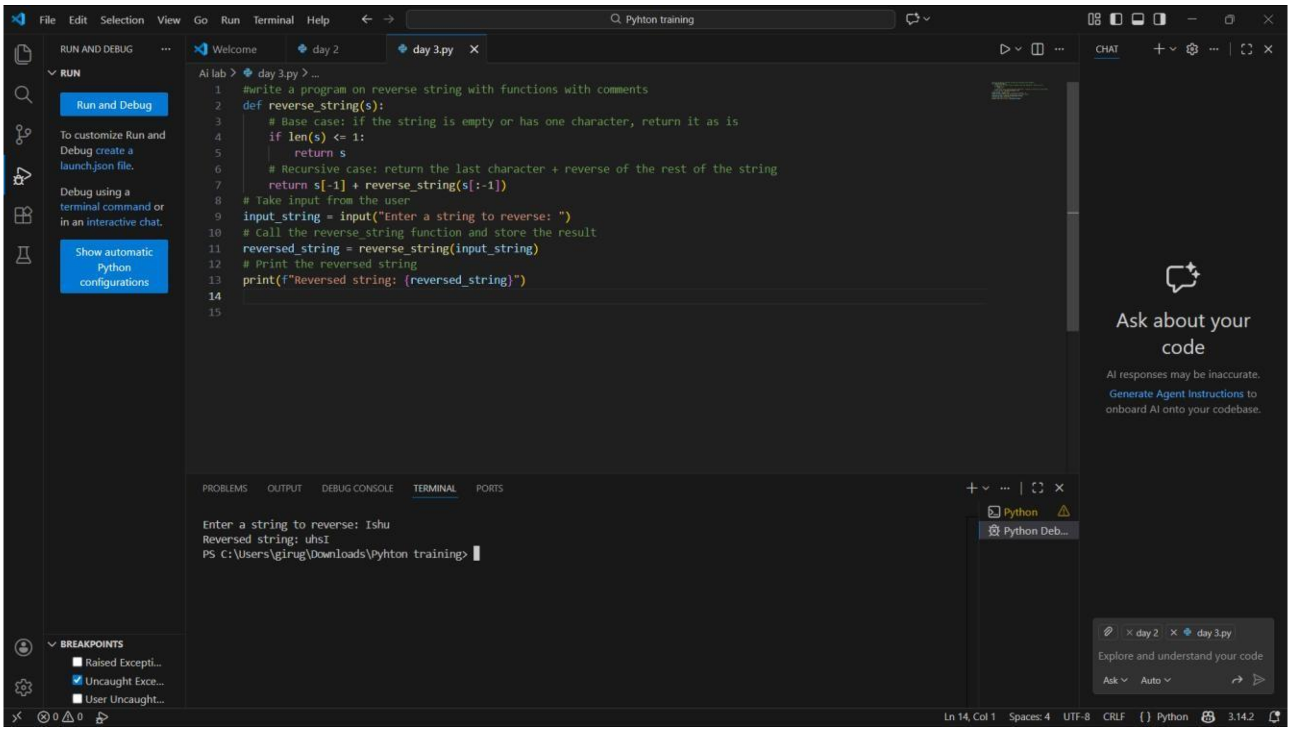
```
Enter a string: harsh
Reversed string: hsrah
PS C:\Users\girug\Downloads\Pyhton training>
```

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”



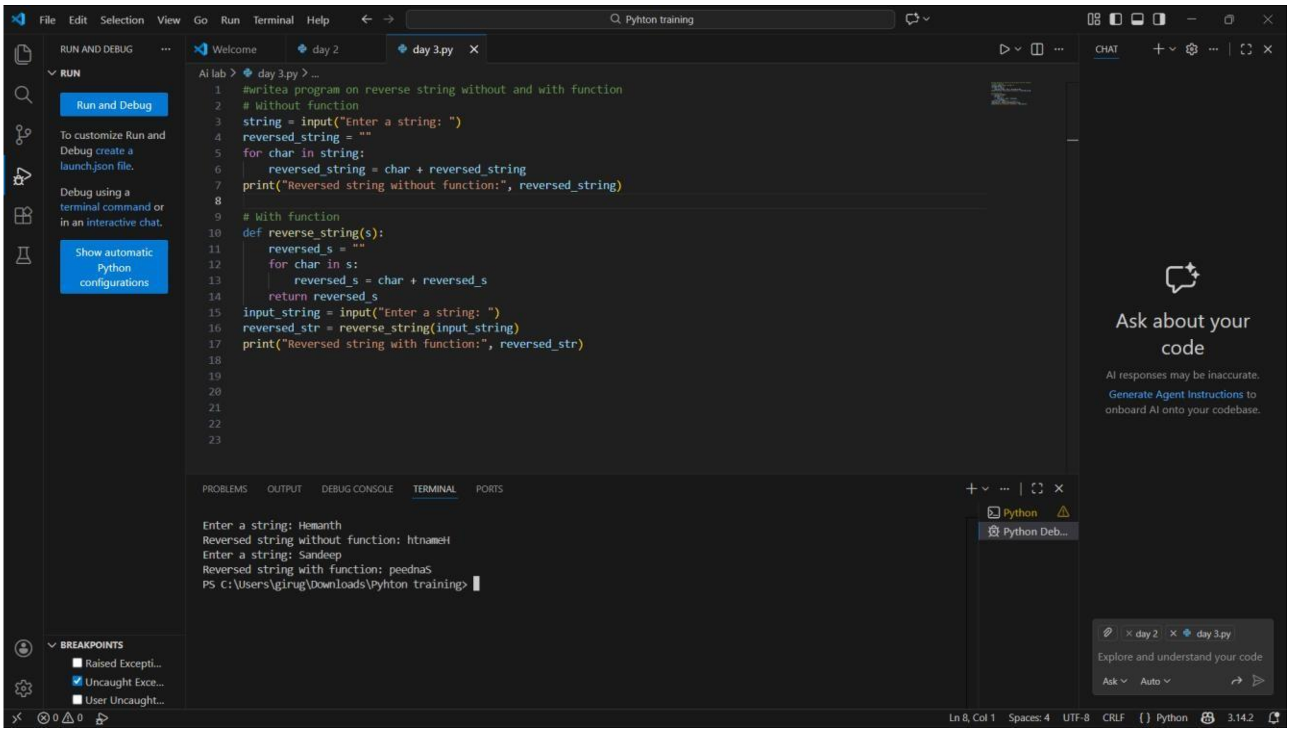
```python
#write a program on reverse string with functions with comments
def reverse_string(s):
    # Base case: if the string is empty or has one character, return it as is
    if len(s) <= 1:
        return s
    # Recursive case: return the last character + reverse of the rest of the string
    return s[-1] + reverse_string(s[:-1])
# Take input from the user
input_string = input("Enter a string to reverse: ")
# Call the reverse_string function and store the result
reversed_string = reverse_string(input_string)
# Print the reversed string
print(f"Reversed string: {reversed_string}")
```

```
Enter a string to reverse: Ishu
Reversed string: uhsI
PS C:\Users\girug\Downloads\Pyhton training>
```

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”



```python
#writea program on reverse string without and with function
# Without function
string = input("Enter a string: ")
reversed_string = ""
for char in string:
    reversed_string = char + reversed_string
print("Reversed string without function:", reversed_string)

# With function
def reverse_string(s):
    reversed_s = ""
    for char in s:
        reversed_s = char + reversed_s
    return reversed_s
input_string = input("Enter a string: ")
reversed_str = reverse_string(input_string)
print("Reversed string with function:", reversed_str)
```

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

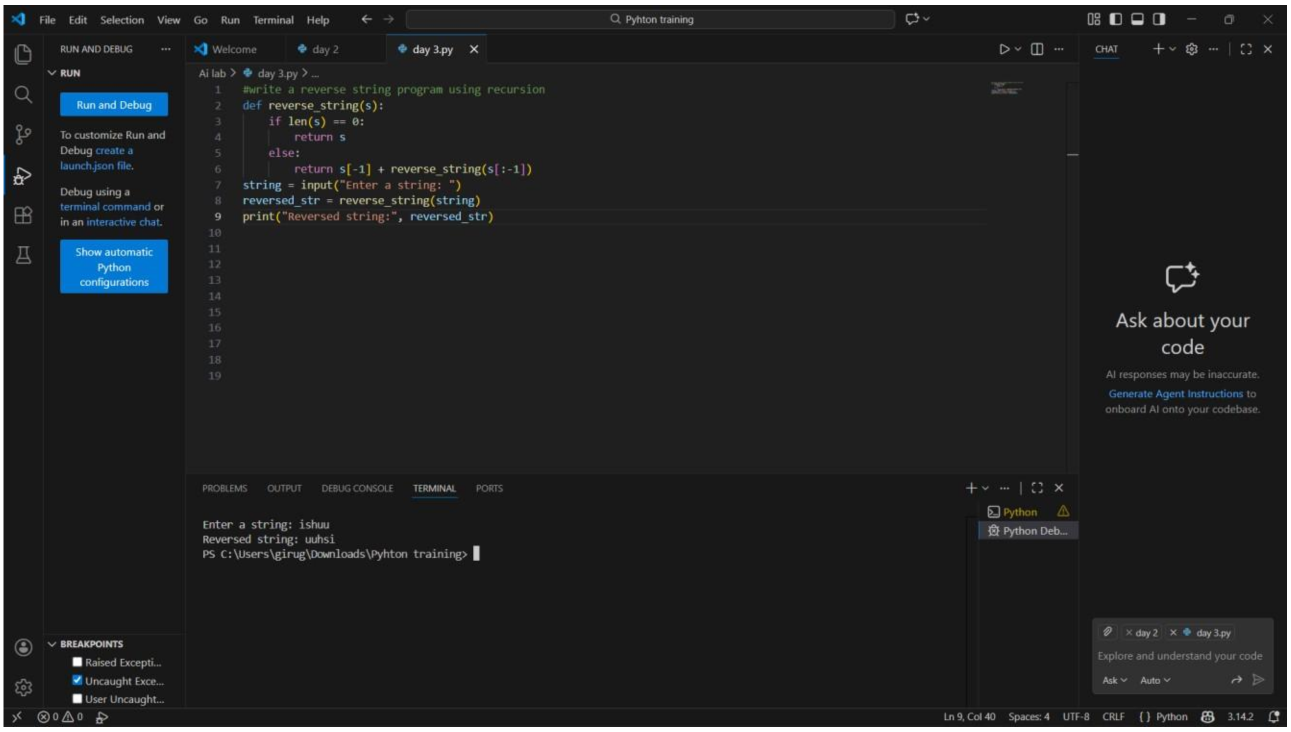
Without using functions: Helps beginners clearly understand the basic logic and step-by-step execution of an Armstrong number program.

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”



File Edit Selection View Go Run Terminal Help
Pyhton training
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug create a launch.json file.
Debug using a terminal command or in an interactive chat.
Show automatic Python configurations
BREAKPOINTS
Raised Excepti...
Uncaught Exce...
User Uncaught...
Welcome
day 2
day 3.py
Ai lab > day 3.py > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Python Deb...
CHAT
Ask about your code
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Explore and understand your code
Ask Auto
Ln 9, Col 40 Spaces: 4 UTF-8 CRLF Python 3.14.2

```python
#write a reverse string program using recursion
def reverse_string(s):
    if len(s) == 0:
        return s
    else:
        return s[-1] + reverse_string(s[:-1])
string = input("Enter a string: ")
reversed_str = reverse_string(string)
print("Reversed string:", reversed_str)
```

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

Execution Flow Explanation

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