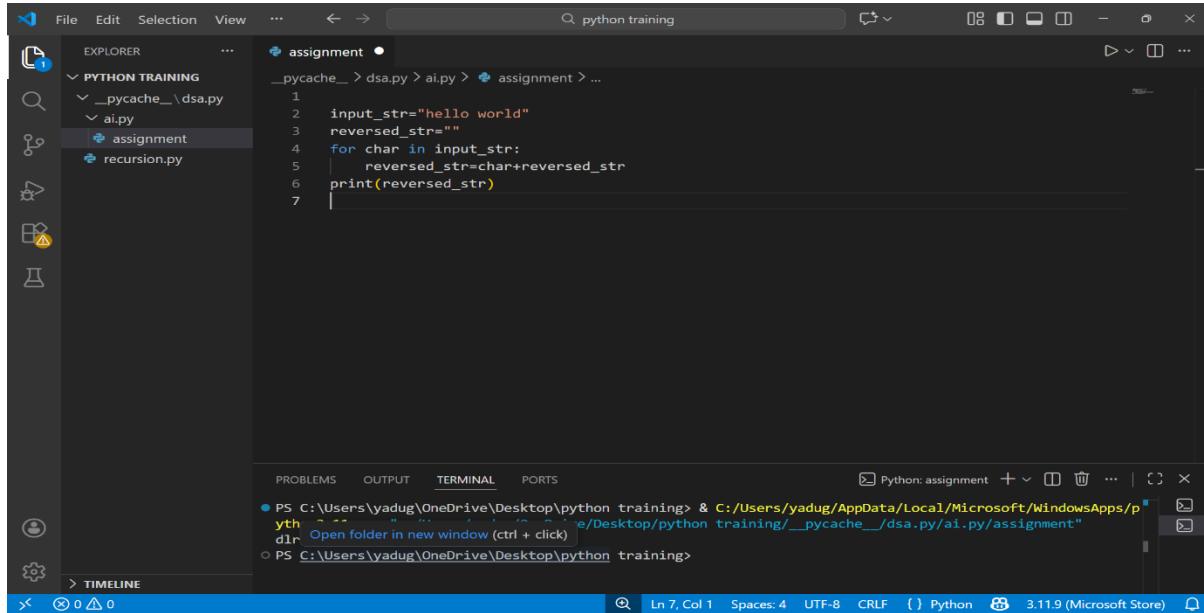


Assignment-1.5

Name: Y.Rajiv

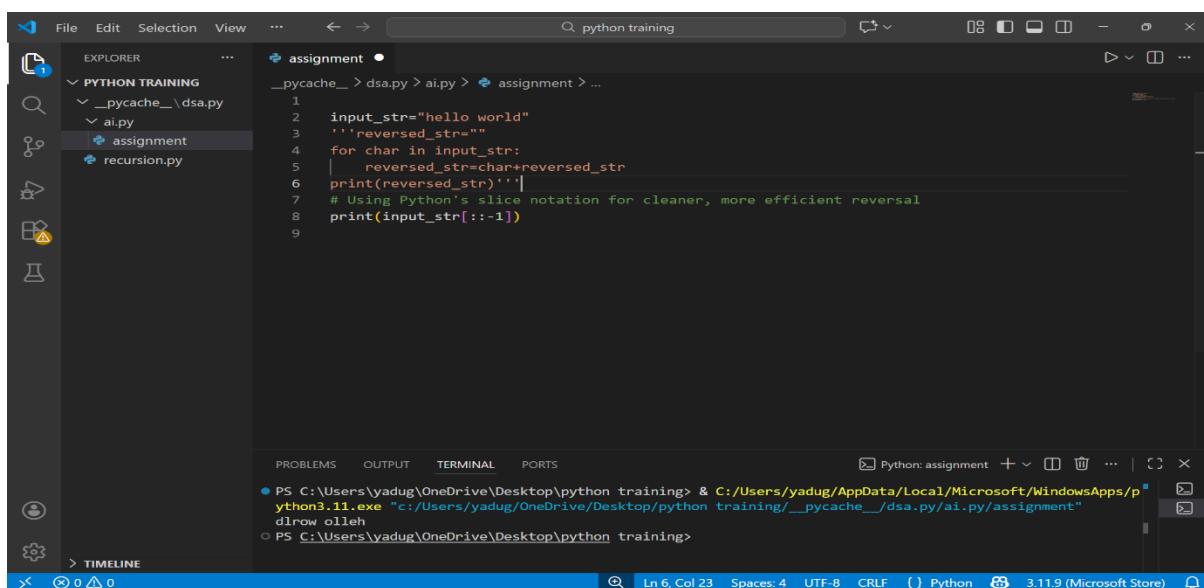
RollNo:2303A51357

TASK-1 AI-Generated Logic Without Modularization (String Reversal Without Functions)



```
File Edit Selection View ... < > python training
EXPLORER PYTHON TRAINING _pycache_ > dsa.py > ai.py > assignment > ...
ai.py
assignment
recursion.py
1 input_str="hello world"
2 reversed_str=""
3 for char in input_str:
4     reversed_str=char+reversed_str
5 print(reversed_str)
6
7
PROBLEMS OUTPUT TERMINAL PORTS
PS C:\Users\yadug\OneDrive\Desktop\python training> & C:/Users/yadug/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/yadug/OneDrive/Desktop/python training/_pycache_/dsa.py/ai.py/assignment"
drlor olleh
PS C:\Users\yadug\OneDrive\Desktop\python training>
Ln 7, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.11.9 (Microsoft Store)
```

TASK-2 Efficiency & Logic Optimization (Readability Improvement)



```
File Edit Selection View ... < > python training
EXPLORER PYTHON TRAINING _pycache_ > dsa.py > ai.py > assignment > ...
ai.py
assignment
recursion.py
1 input_str="hello world"
2 '''reversed_str=""
3 for char in input_str:
4     reversed_str=char+reversed_str
5 print(reversed_str)'''
6 # Using Python's slice notation for cleaner, more efficient reversal
7 print(input_str[::-1])
8
9
PROBLEMS OUTPUT TERMINAL PORTS
PS C:\Users\yadug\OneDrive\Desktop\python training> & C:/Users/yadug/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/yadug/OneDrive/Desktop/python training/_pycache_/dsa.py/ai.py/assignment"
olleh dlrow
PS C:\Users\yadug\OneDrive\Desktop\python training>
Ln 6, Col 23 Spaces: 4 UTF-8 CRLF {} Python 3.11.9 (Microsoft Store)
```

- Explanation: `[::-1]` tells Python:
 - Start from end

- Move backwards
- Step = -1
- Python internally reverses the string in **one pass**.

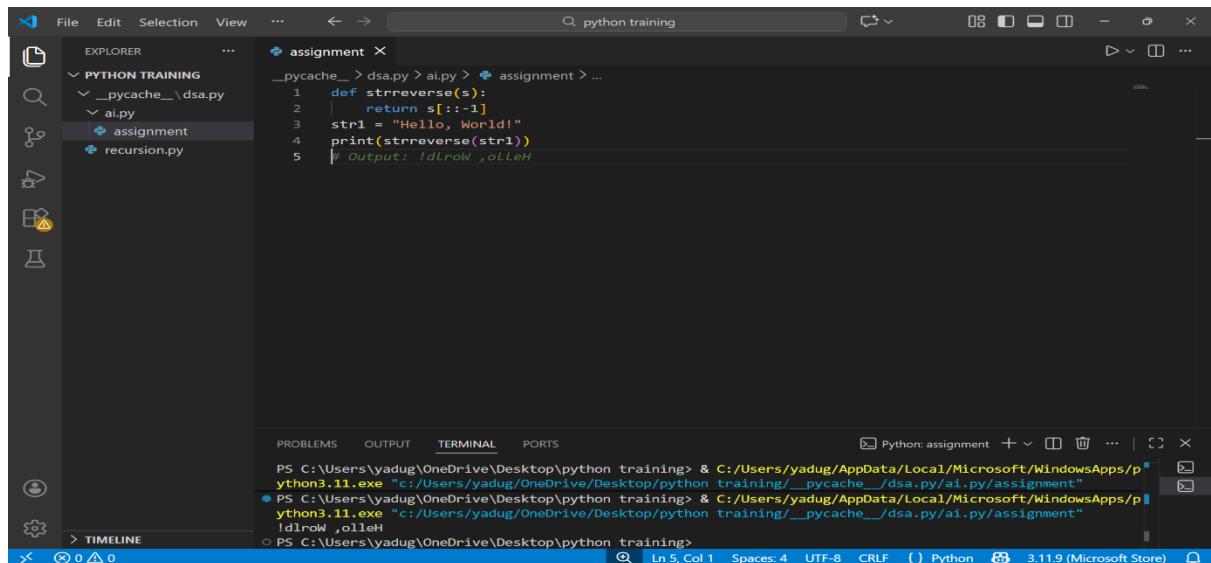
Time Complexity

- Each character is accessed **once**.
- No repeated copying like the loop version.

Time Complexity: O(n)

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

Before optimization



```
def strreverse(s):
    return s[::-1]
str1 = "Hello, World!"
print(strreverse(str1))
# Output: !dlroW ,olleH
```

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left lists files: 'ai.py' and 'recursion.py'. The 'ai.py' file is open in the editor, displaying the code above. Below the editor is the 'TERMINAL' tab, which shows the command line output of running the script. The terminal window title is 'Python: assignment'. The output shows the string 'Hello, World!' being reversed to '!dlroW ,olleH'.

After optimization using copilot

```

assignment
def strreverse(s):
    """Reverses the string."""
    return s[::-1]
strreverse("hello") # Example usage; returns "olleH"

```

PROBLEMS OUTPUT TERMINAL PORTS

```

PS C:\Users\yadug\OneDrive\Desktop\python training> & C:/Users/yadug/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/yadug/OneDrive/Desktop/python training/_pycache_/dsa.py/assignment"
PS C:\Users\yadug\OneDrive\Desktop\python training> & C:/Users/yadug/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/yadug/OneDrive/Desktop/python training/_pycache_/dsa.py/ai.py/assignment"
idleW ,olleH
PS C:\Users\yadug\OneDrive\Desktop\python training>

```

Ln 4, Col 21 Spaces: 4 UTF-8 CRLF Python 3.11.9 (Microsoft Store)

Task-4 Comparative Analysis – Procedural vs Modular Approach (With vs Without Functions)

Criteria	Without Functions (Procedural)	With Functions (Modular)
Code Clarity	Logic mixed with execution, harder to read in large files	Clear separation of logic and execution
Reusability	Code must be rewritten for every use	Function can be reused anywhere
Debugging Ease	Bugs are harder to isolate	Easy to debug by testing the function alone
Scalability	Poor for large programs	Highly suitable for large applications
Maintenance	Changes must be made in multiple places	Changes made once in the function
Testing	Manual testing only	Supports unit testing easily
Readability	Acceptable for small scripts	Excellent for professional codebases
Copilot Assistance	Limited suggestions	Strong AI support for docstrings & logic

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

The screenshot shows the VS Code interface with the following details:

- Explorer:** Shows a folder structure under "PYTHON TRAINING" containing "assignment" and "recursion.py".
- Code Editor:** The "assignment" tab is active, displaying Python code for string reversal:

```
1 # Loop-based string reversal
2
3 # Method 1: Using a for loop with reversed index
4 def reverse_string_1(s):
5     reversed_str = ""
6     for i in range(len(s) - 1, -1, -1):
7         reversed_str += s[i]
8     return reversed_str
9 reverse_string_1("hello") #output: "olleh"
10 def reverse_string_slice(text):
11     return text[::-1]
```
- Terminal:** Shows the command line output of running assignment.py:

```
PS C:\Users\yadug\OneDrive\Desktop\python training> & C:/Users/yadug/AppData/Local/Microsoft/WindowsApps/python3
.11.exe "c:/Users/yadug/OneDrive/Desktop/python training/_pycache__dsa.py/ai.py/assignment"
PS C:\Users\yadug\OneDrive\Desktop\python training>
```
- Status Bar:** Shows the file is "Python: assignment", line 11, column 5, spaces: 4, encoding: UTF-8, CRLF, Python 3.11.9 (Microsoft Store).

Explanation:

Both approaches correctly reverse strings.

Slicing-based reversal is superior in performance, readability, and scalability.