SR UNIVERSITY

School of Computer Science and Artificial Intelligence

Lab Assignment # 1

Program	**: B. Tech (CSE)**
Specialization	**: CSE**
Course Title	**: AI Assisted Coding**
Course Code	**: 23CS201PC302**
Semester	**: 3 -2**
Academic Session	**: 2025-2026**
Name of Student	**: Eshwar**
Enrollment No.	**: 2403a51l26**
Batch No.	**: 51**
Date	**:20-01-2026**

Submission Instructions:

(All instructions should be followed strictly to avoid deduction of marks)

1. Use the same file to complete the assignment and don't change the settings.
2. Minimum 10 screen shots of your account should be taken to showcase your work.
3. **File Format:**
   - Submit your assignment as a PDF document (pdf). Ensure the file is named according to the following convention:

     `BNo_StudentName_AI_Coding_A1.`

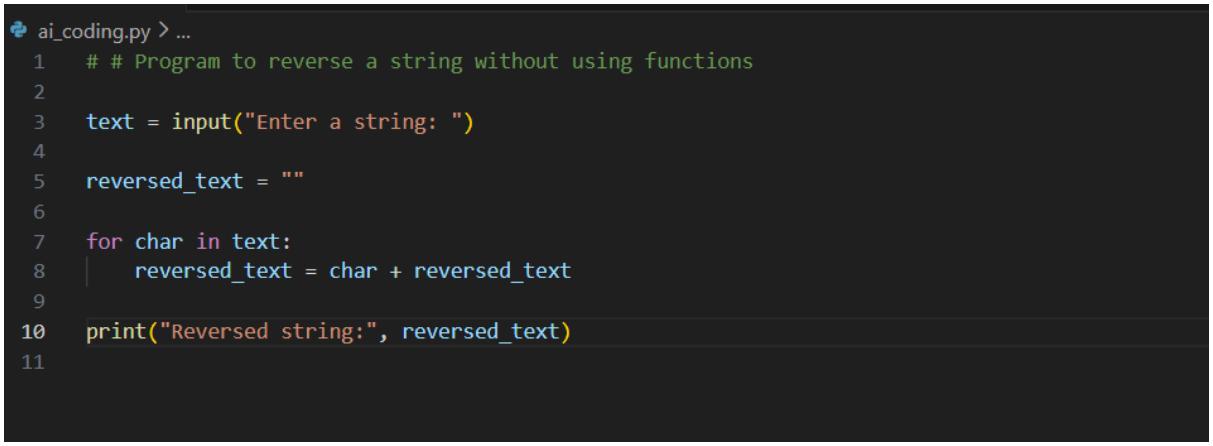
     `Sample`: **B10_Rohit_22A523421_A1**
4. Fill all the entries mentioned on top section.
5. Mention your AWS Academy Virtual Lab Account details as shown in the next page.
6. **Don't write on this page.**
7. All answers should be answered from next page only.

Lab assignment: 1.5

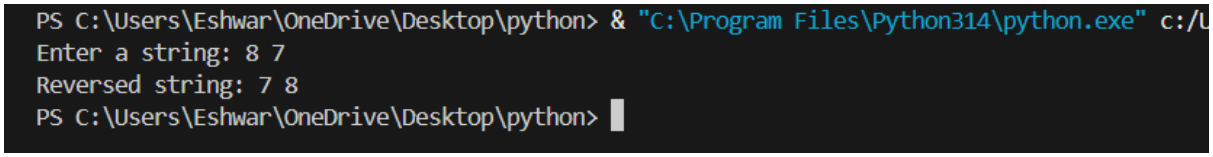
Task 1:

Prompt:

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



Output:



- The program first takes a string input from the user.
- It then uses a loop to read each character and adds it in reverse order to a new string.
- Finally, the reversed string is printed as output.

Task 2:

Prompt:

Efficiency & Logic Optimization (Readability Improvement):

```
ai_coding.py > ...
1   #  # String reversal without functions
2
3   text = input("Enter a string: ")
4
5   reversed_text = ""
6
7   for char in text:
8       reversed_text = char + reversed_text
9
10  print("Reversed string:", reversed_text)
```

Output:

```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c
Enter a string: 6 9
Reversed string:  9 6
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

Optimized Code (Improved Readability & Efficiency):

```python
#  # Optimized string reversal

text = input("Enter a string: ")
reversed_text = text[::-1]

print("Reversed string:", reversed_text)

```

Output:

```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:/Us
Enter a string: 9 3
Reversed string: 3 9
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

- **The program takes a string as input from the user.**
- **It reverses the string using Python's slicing method [::-1], which is fast and efficient.**
- **The reversed string is then displayed as output in a clean and readable way.**

Task 3 :

prompt:

Modular Design Using AI Assistance (String Reversal Using Functions):

ai_coding.py > ...

```python
#  # Function to reverse a string
def reverse_string(text):
    """
    This function takes a string as input
    and returns the reversed version of it.
    """
    return text[::-1]


# Main program
user_input = input("Enter a string: ")
result = reverse_string(user_input)

print("Reversed string:", result)

```

Output:

```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe"
Enter a string: 5 7 3 7
Reversed string: 7 3 7 5
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

- **The program defines a function that reverses a string using slicing.**
- **The function returns the reversed string to the main program.**
- **This modular approach improves reusability, readability, and maintainability.**

Task 4:

Prompt:

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

ai_coding.py > ...

```python
#  # String reversal without using functions

text = input("Enter a string: ")

reversed_text = ""

for char in text:
    reversed_text = char + reversed_text

print("Reversed string:", reversed_text)
```

Output:

```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe
Enter a string: 98 56 89
Reversed string: 98 65 89
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

With Functions (Modular Approach):

ai_coding.py > ...

```python
# # String reversal using a function

def reverse_string(text):
    return text[::-1]

user_input = input("Enter a string: ")
result = reverse_string(user_input)

print("Reversed string:", result)
```

Output:

```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe
Enter a string: 55 77 33 66
Reversed string: 66 33 77 55
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

- **The program takes a string as input from the user.**

- **It reverses the string either directly (procedural) or using a function (modular).**

- **The modular approach improves code readability, reusability, and maintenance.**

Task 5:

Prompt:

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

```
ai_coding.py > ...
1   ## Iterative string reversal
2
3   text = input("Enter a string: ")
4
5   reversed_text = ""
6   for char in text:
7       reversed_text = char + reversed_text
8
9   print("Reversed string:", reversed_text)
10
```

Output:

```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:/Users/E
Enter a string: 21 67 90
Reversed string:  09 76 12
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

Built-in / Slicing-Based String Reversal:

ai_coding.py > ...

```python
# # Built-in string reversal using slicing

text = input("Enter a string: ")
reversed_text = text[::-1]

print("Reversed string:", reversed_text)
```

Output:

```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.ex
Enter a string: 4 7 6 9 3
Reversed string: 3 9 6 7 4
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

- **The iterative method reverses a string using a loop, adding characters one by one in reverse order.**
- **The slicing method reverses the string in a single step using Python's built-in [::-1] syntax.**
- **Slicing is faster, cleaner, and more efficient, while the iterative approach helps understand the underlying logic.**