

# ASSIGNMENT – 1

AI ASSISTANT CODING

2303A510D9

BT-29

TASK – 1

PROMPT:

String Reversal Without Functions

```
#generate string reversal using without function
def reverse_string(s):
    reversed_str = ""
    for char in s:
        reversed_str = char + reversed_str
    return reversed_str

input_string = "hello"
print("Original string:", input_string)
print("Reversed string:", reverse_string(input_string))
```

Output :

```
Original string: hello
Reversed string: olleh
PS C:\Users\gunda\AI ASSIST CODING - 1>
```

**OBSERVATION - TASK 1** This program reverses a string without using any built-in function by adding each character to the front of a new string. It processes the string character by character and finally prints the original and reversed string.

## TASK -2

Prompt :-

“Simplify this string reversal code”

“Improve readability and efficiency

```
# String reversal without using any function
input_string = "hello"
reversed_str = ""
for char in input_string:
    reversed_str = char + reversed_str
print("Original string:", input_string)
print("Reversed string:", reversed_str)
#output: !dlrow ,olleh
```

## OUTPUT:

```
Original string: hello
Reversed string: olleh
PS C:\Users\gunda\AI ASSIST CODING - 1>
```

**OBSERVATION - TASK 2** The program takes a string as input and reverses it without using any built-in function. It displays both the original string and the reversed string as the output.

### **TASK – 3**

**Prompt :-**

**String Reversal Using Functions**

```
#generate a string reversal using function
num = 5
factorial = 1
for i in range(1, num + 1):
    factorial *= i
```

**OUTPUT:**

```
The factorial of 5 is: 120
PS C:\Users\gunda\AI ASSIST CODING - 1>
```

**OBSERVATION - TASK 3** The program calculates the factorial of a given number using a loop and displays the result. For the input value 5, it computes and prints the factorial as 120.

**TASK - 4**

Criteria	Procedural (Task 1)	Modular (Task 3)
Code Clarity	Low; logic is mixed with I/O.	High; logic is separated from I/O.
Reusability	Zero; requires copy-pasting.	High; function can be imported
Debugging	Harder to test specific logic.	Easy to unit test the function.
Scalability	Poor for large systems.	Essential for large systems

**TASK- 5**

```
AI-generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches to String Reversal)
def fibonacci_reverse(s):
    n = len(s)
    fib = [0, 1]
    for i in range(2, n + 1):
        fib.append(fib[-1] + fib[-2])

    reversed_str = ""
    for i in range(n - 1, -1, -1):
        reversed_str += s[i]
    return reversed_str

test_string = "hello"
print("Fibonacci approach - Original:", test_string)
print("Fibonacci approach - Reversed:", fibonacci_reverse(test_string))
```

## OUTPUT :

```
fibonacci approach - Original: hello  
fibonacci approach - Reversed: olleh  
PS C:\Users\gunda\AI ASSIST CODING - 1>
```

**OBSERVATION- TASK 5** The program uses a custom function to reverse a given string and displays both the original and reversed versions. For the input "hello", it prints the reversed string as "olleh"