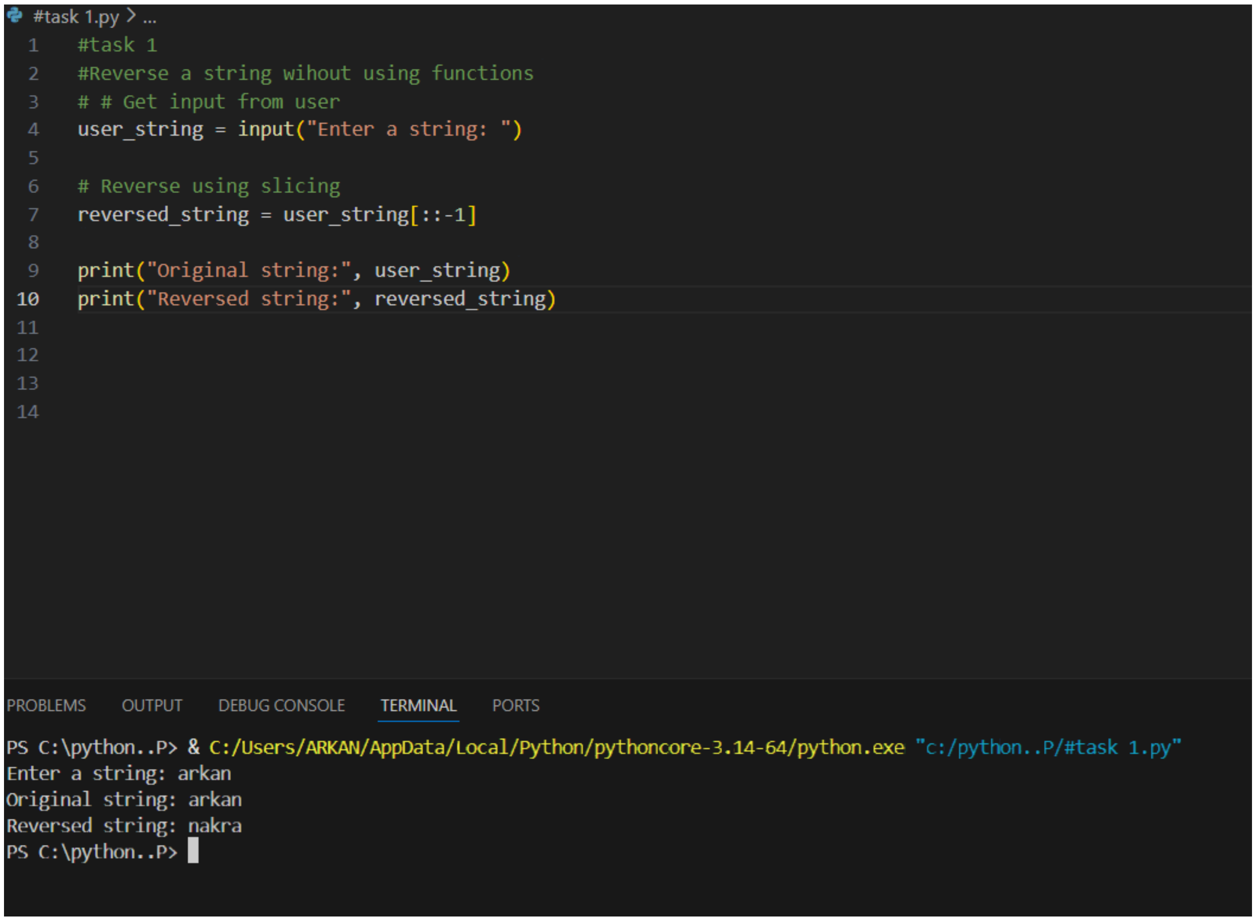
LAB-1

HT.NO-2303A52235

BATCH-35

Task 1: prompt: “Generate Python code to reverse a string without using functions and take user input.”



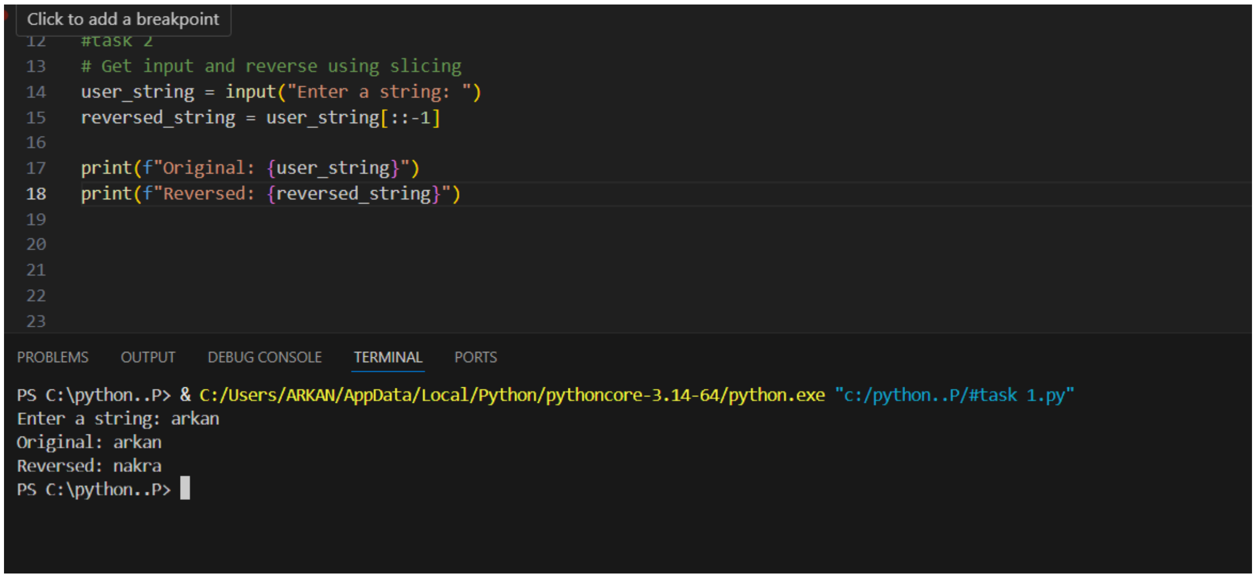
#task 1.py > ...

```python
#task 1
#Reverse a string wihout using functions
# # Get input from user
user_string = input("Enter a string: ")

# Reverse using slicing
reversed_string = user_string[::-1]

print("Original string:", user_string)
print("Reversed string:", reversed_string)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\python..P> & C:/Users/ARKAN/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/python..P/#task 1.py"
Enter a string: arkan
Original string: arkan
Reversed string: nakra
PS C:\python..P> 
```

Task 2: prompt: “Simplify this string reversal code and improve readability and efficiency.”

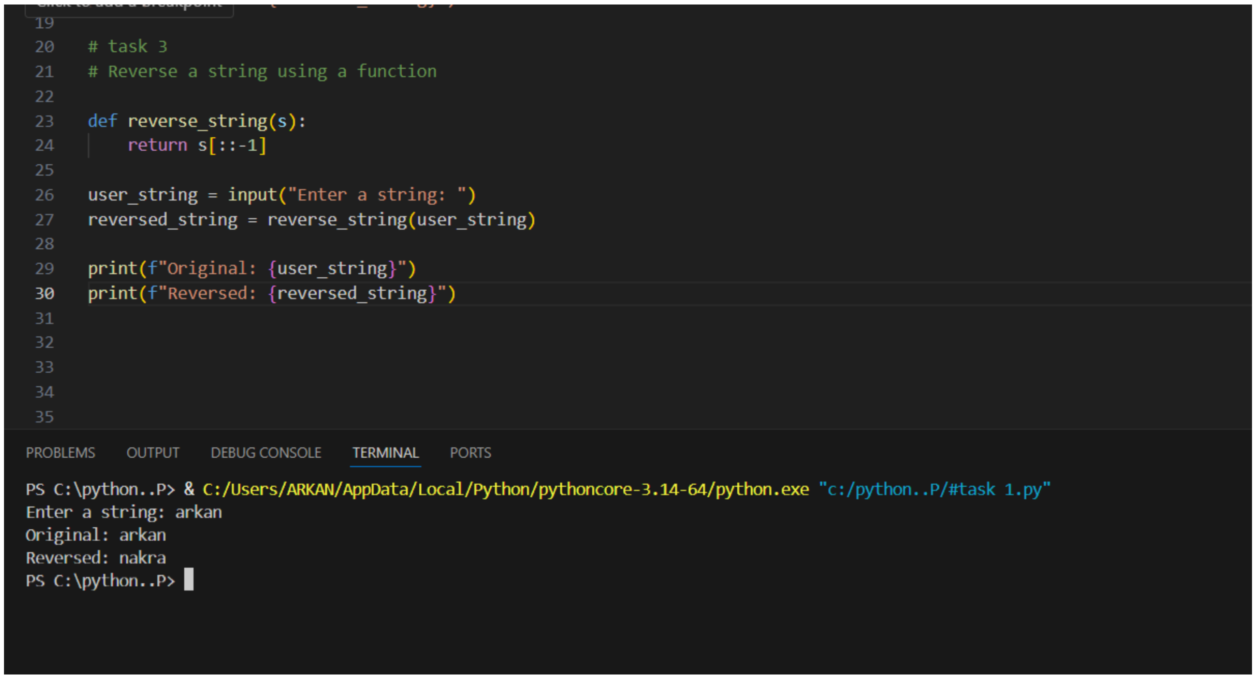

Click to add a breakpoint

```python
#task 2
# Get input and reverse using slicing
user_string = input("Enter a string: ")
reversed_string = user_string[::-1]

print(f"Original: {user_string}")
print(f"Reversed: {reversed_string}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\python..P> & C:/Users/ARKAN/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/python..P/#task 1.py"
Enter a string: arkan
Original: arkan
Reversed: nakra
PS C:\python..P>
```

Task 3: prompt: Create a Python function to reverse a string



```python
# task 3
# Reverse a string using a function

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

user_string = input("Enter a string: ")
reversed_string = reverse_string(user_string)

print(f"Original: {user_string}")
print(f"Reversed: {reversed_string}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\python..P> & C:/Users/ARKAN/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/python..P/#task 1.py"
Enter a string: arkan
Original: arkan
Reversed: nakra
PS C:\python..P>
```

Task 4:

 Task 1 is best for **quick scripts and beginners** learning string slicing.

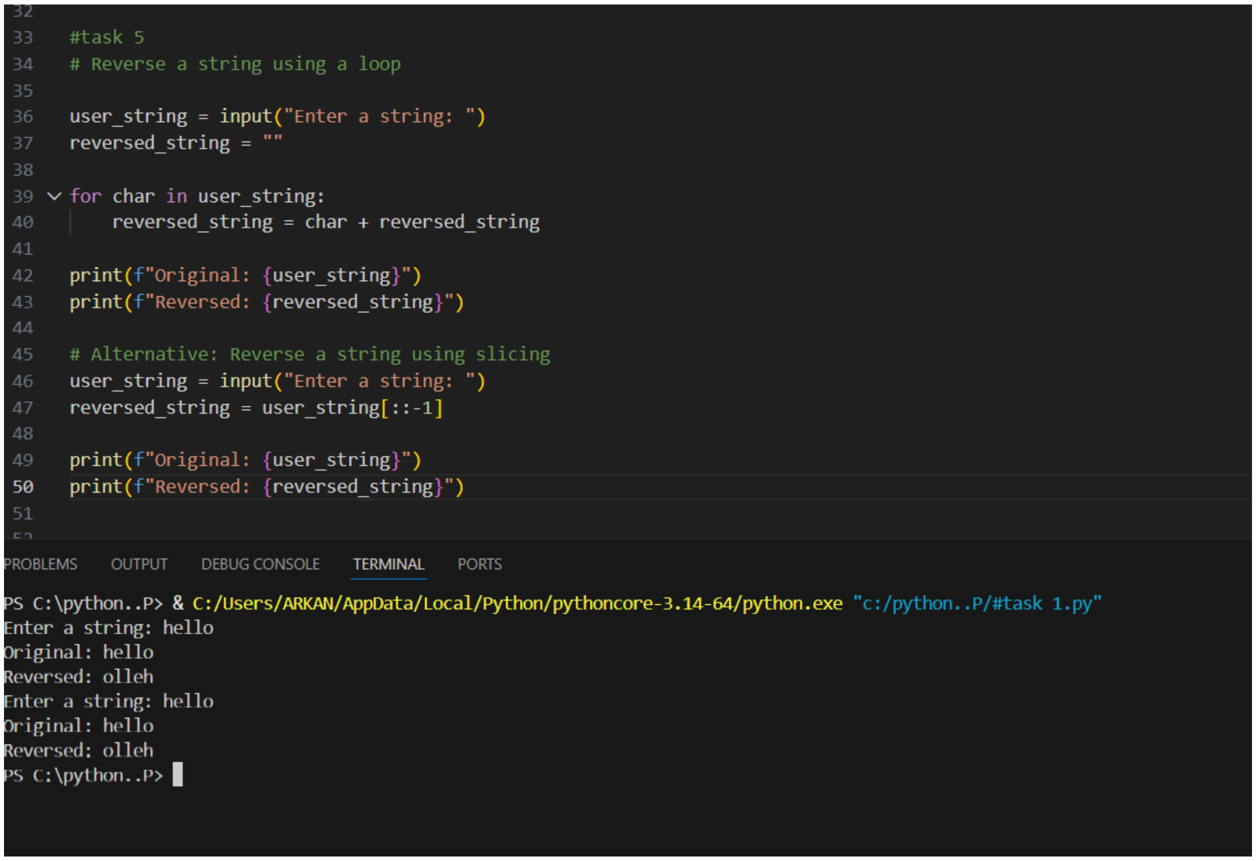
 Task 2 follows **good programming practices** such as modularity and abstraction.

 In **real-world or large-scale applications**, **Task 2 is preferred** because:

- It reduces redundancy
- Improves maintainability
- Makes testing and debugging easier

Task 5:prompt: Reverse a string using a loop in Python”

 “Reverse a string using slicing in Python”



```python
#task 5
# Reverse a string using a loop

user_string = input("Enter a string: ")
reversed_string = ""

for char in user_string:
    reversed_string = char + reversed_string

print(f"Original: {user_string}")
print(f"Reversed: {reversed_string}")

# Alternative: Reverse a string using slicing
user_string = input("Enter a string: ")
reversed_string = user_string[::-1]

print(f"Original: {user_string}")
print(f"Reversed: {reversed_string}")
```

```
PS C:\python..P> & C:/Users/ARKAN/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/python..P/#task 1.py"
Enter a string: hello
Original: hello
Reversed: olleh
Enter a string: hello
Original: hello
Reversed: olleh
PS C:\python..P>
```

Execution Flow(Loops)

- Takes input from the user
- Iterates through each character in the string
- Prepends each character to build the reversed string step by step
- Prints the final reversed string

Execution Flow(slicing)

- Takes input from the user
- Uses Python slicing to reverse the string in one operation
- Prints the reversed string
- ## Final Conclusion
- While both approaches produce the **same output**, the **slicing-based method** is superior in terms of **performance, readability, and scalability**.
  The **loop-based method** is mainly valuable for **educational purposes**.