Lab Assignment -1

Name : Gaddam Padmavathi

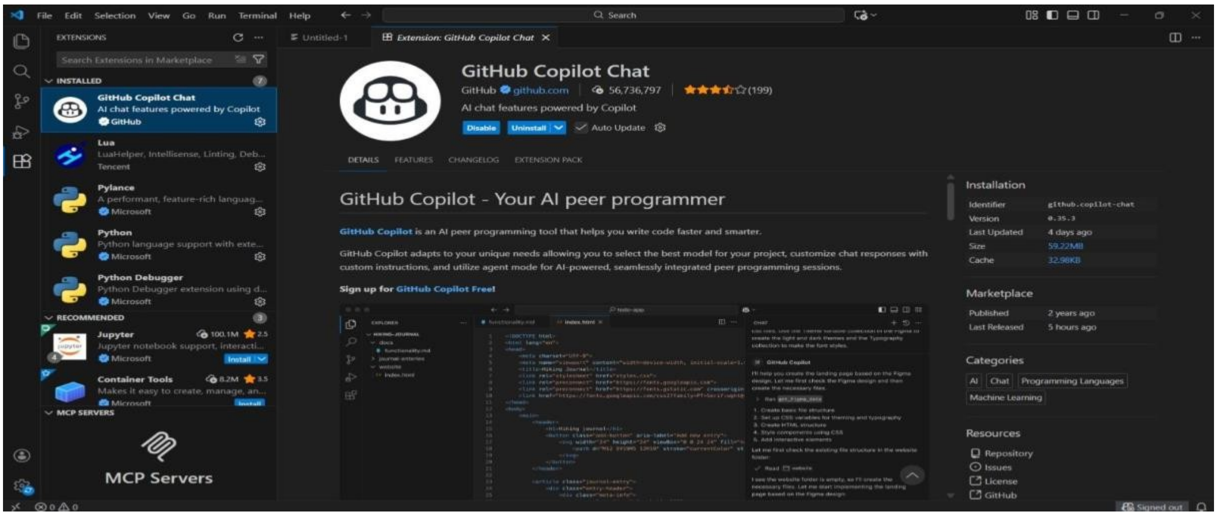
HT.No : 2303A52269

Batch : 36

Environment Setup – *GitHub Copilot and VS Code Integration + Understanding AI-assisted Coding Workflow*

Task 0

Install and configure GitHub Copilot in VS Code. Take screenshots of each step.



Extension: GitHub Copilot Chat
GitHub Copilot Chat
GitHub github.com | 56,736,797 | (199)
AI chat features powered by Copilot
Disable
Uninstall
Auto Update
DETAILS
FEATURES
CHANGELOG
EXTENSION PACK
MCP Servers

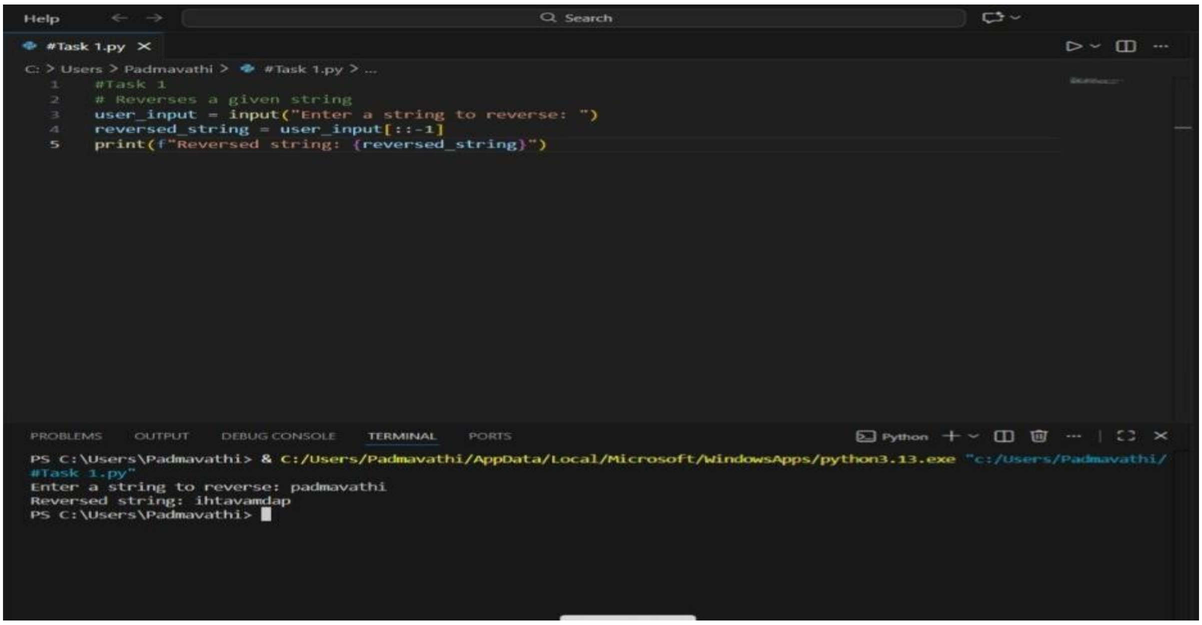
GitHub Copilot - Your AI peer programmer

GitHub Copilot is an AI peer programming tool that helps you write code faster and smarter.

GitHub Copilot adapts to your unique needs allowing you to select the best model for your project, customize chat responses with custom instructions, and utilize agent mode for AI-powered, seamlessly integrated peer programming sessions.

Sign up for GitHub Copilot Free!

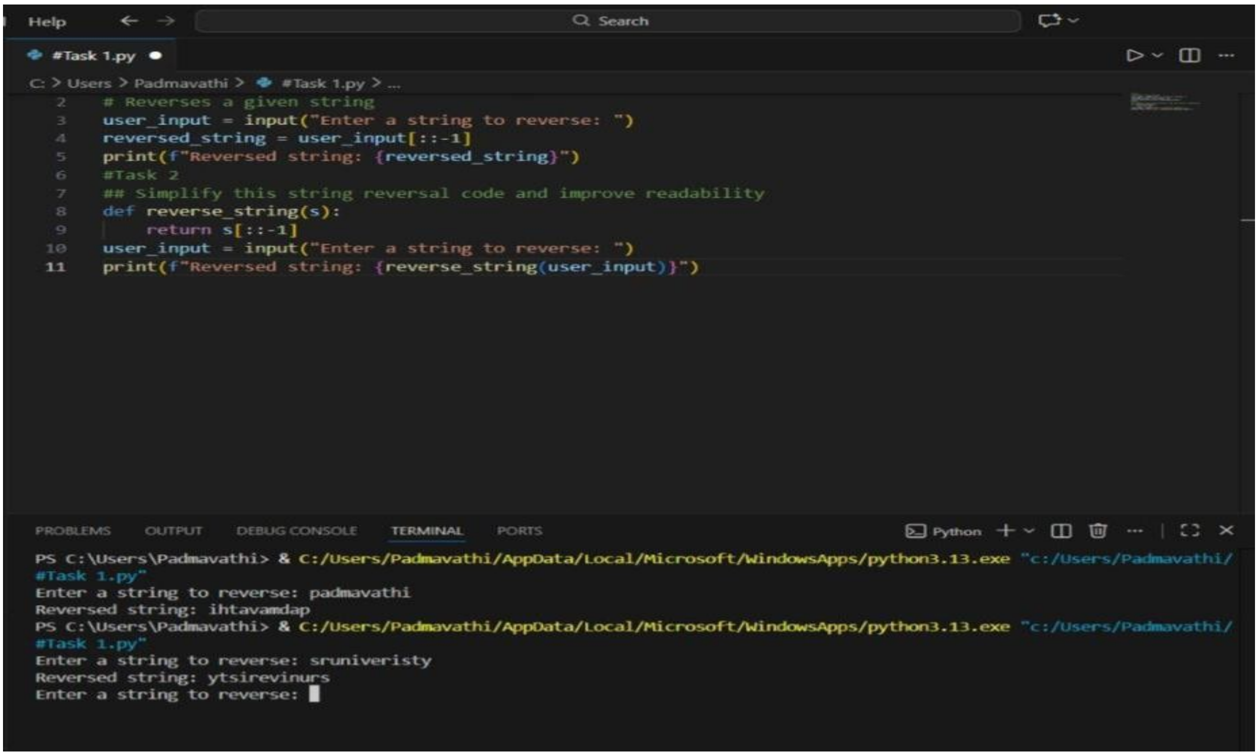
Task 1: AI-Generated Logic Without Modularization (String Reversal Without Functions)



```python
#Task 1
# Reverses a given string
user_input = input("Enter a string to reverse: ")
reversed_string = user_input[::-1]
print(f"Reversed string: {reversed_string}")
```

```
PS C:\Users\Padmavathi> & C:/Users/Padmavathi/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/Padmavathi/
#Task 1.py"
Enter a string to reverse: padmavathi
Reversed string: ihtavamdap
PS C:\Users\Padmavathi> █
```

Task 2: Efficiency & Logic Optimization (Readability Improvement)

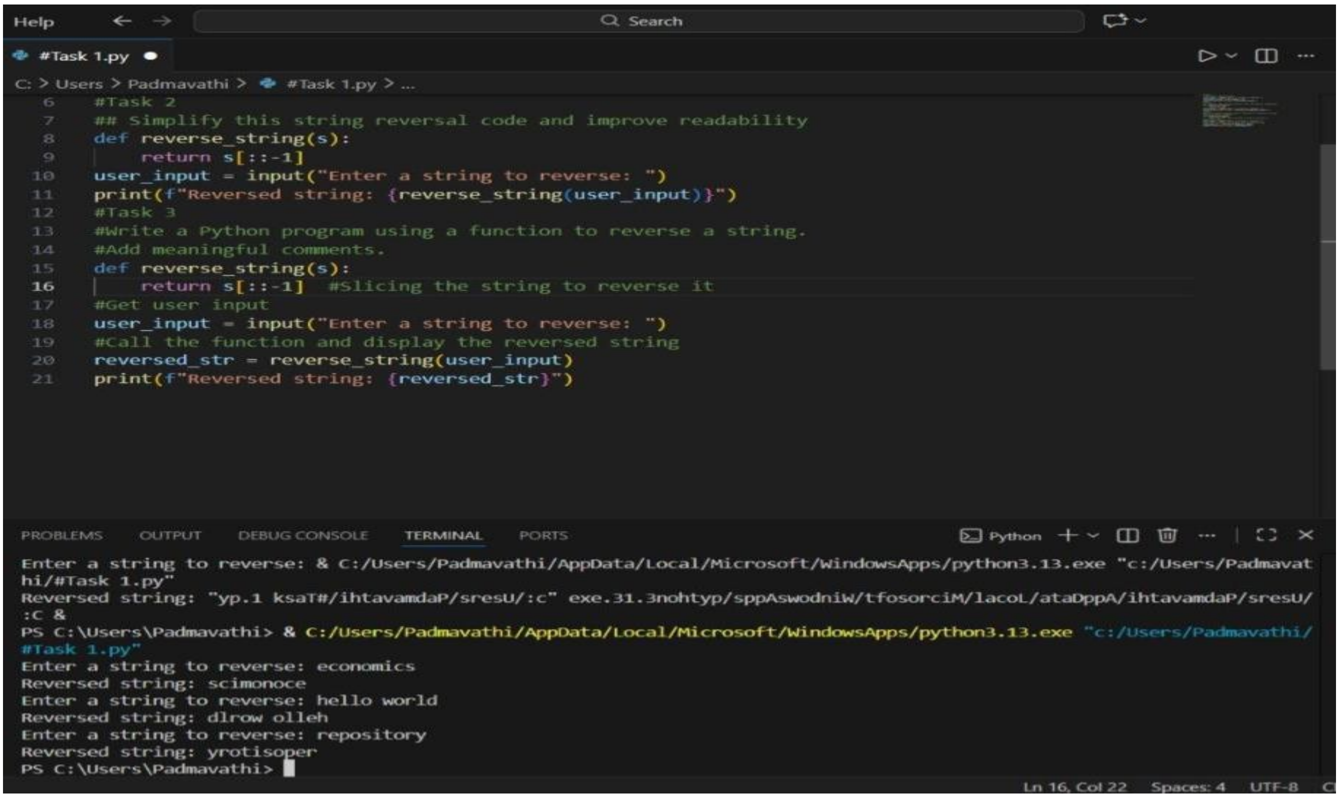


```python
# Reverses a given string
user_input = input("Enter a string to reverse: ")
reversed_string = user_input[::-1]
print(f"Reversed string: {reversed_string}")
#Task 2
## Simplify this string reversal code and improve readability
def reverse_string(s):
    return s[::-1]
user_input = input("Enter a string to reverse: ")
print(f"Reversed string: {reverse_string(user_input)}")
```

```
PS C:\Users\Padmavathi> & C:/Users/Padmavathi/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/Padmavathi/
#Task 1.py"
Enter a string to reverse: padmavathi
Reversed string: ihtavamdap
PS C:\Users\Padmavathi> & C:/Users/Padmavathi/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/Padmavathi/
#Task 1.py"
Enter a string to reverse: sruniveristy
Reversed string: ytsirevinurs
Enter a string to reverse: 
```

Explanation : In Task 2, the original string reversal code was optimized by removing unnecessary variables and simplifying the logic.
The loop and intermediate variable were eliminated, and Python slicing was used directly.
This improved readability and reduced code length.
Both versions have a time complexity of O(n), but the optimized version is more efficient in practice due to fewer operations.

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



```python
#Task 2
## Simplify this string reversal code and improve readability
def reverse_string(s):
    return s[::-1]
user_input = input("Enter a string to reverse: ")
print(f"Reversed string: {reverse_string(user_input)}")
#Task 3
#Write a Python program using a function to reverse a string.
#Add meaningful comments.
def reverse_string(s):
    return s[::-1]  #Slicing the string to reverse it
#Get user input
user_input = input("Enter a string to reverse: ")
#Call the function and display the reversed string
reversed_str = reverse_string(user_input)
print(f"Reversed string: {reversed_str}")
```

```
Enter a string to reverse: & C:/Users/Padmavathi/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/Padmavat
hi/#Task 1.py"
Reversed string: "yp.1 ksaT#/ihtavamdaP/sresU/:c" exe.31.3nohtyp/sppAswodniW/tfosorciM/lacoL/ataDppA/ihtavamdaP/sresU/
:C &
PS C:\Users\Padmavathi> & C:/Users/Padmavathi/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/Padmavathi/
#Task 1.py"
Enter a string to reverse: economics
Reversed string: scimonoce
Enter a string to reverse: hello world
Reversed string: dlrow olleh
Enter a string to reverse: repository
Reversed string: yrotisoper
PS C:\Users\Padmavathi>
```

Explanation

- The function reverse_string() encapsulates the string reversal logic
- It uses Python slicing ([::-1]) for efficient reversal
- The function returns the reversed string to the caller
- This modular approach allows reuse of the same logic in multiple parts of an application
- Meaningful comments improve code readability and understanding

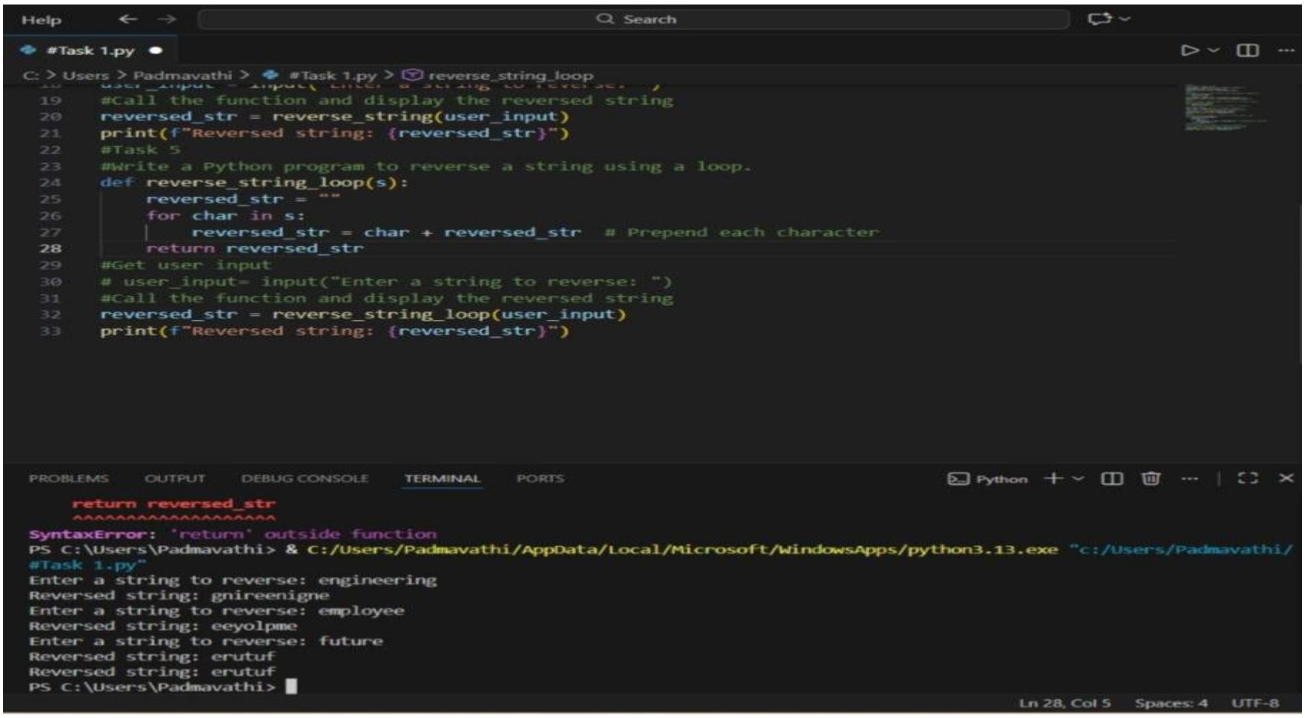
Task 4: Comparative Analysis – Procedural vs Modular Approach (With vs Without Functions)

Comparison Table

Criteria	Without Functions (Procedural)	With Functions (Modular)
Code Clarity	Logic is mixed with input/output, making it less clear	Logic is separated into a function, improving clarity
Reusability	Code cannot be reused easily	Function can be reused in multiple parts of the application
Debugging Ease	Harder to debug due to lack of separation	Easier to debug and test individual functions
Maintainability	Changes must be made in multiple places	Changes can be made in one function
Scalability	Not suitable for large programs	Suitable for large-scale applications

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

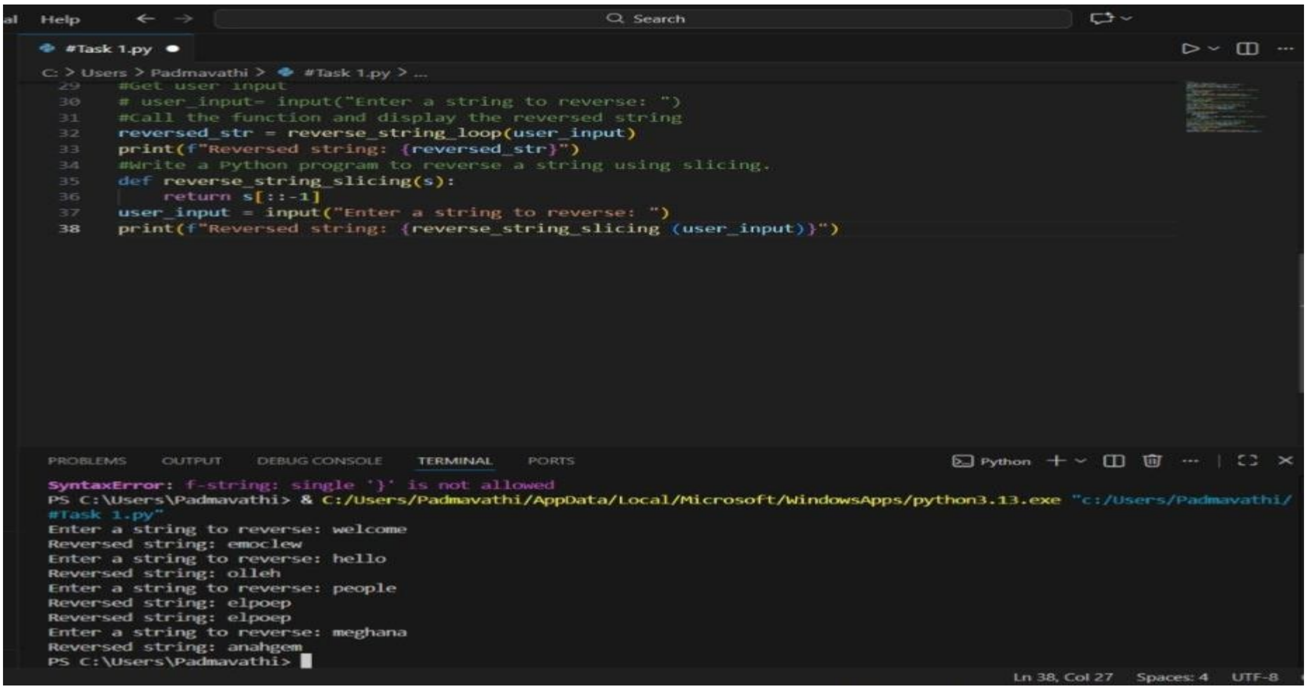
□ A **loop-based** string reversal approach



```python
#Call the function and display the reversed string
reversed_str = reverse_string(user_input)
print(f"Reversed string: {reversed_str}")
#Task 5
#Write a Python program to reverse a string using a loop.
def reverse_string_loop(s):
    reversed_str = ""
    for char in s:
        reversed_str = char + reversed_str  # Prepend each character
    return reversed_str
#Get user input
# user_input= input("Enter a string to reverse: ")
#Call the function and display the reversed string
reversed_str = reverse_string_loop(user_input)
print(f"Reversed string: {reversed_str}")
```

```
    return reversed_str
    ^^^^^^^^^^^^^^^^^^^
SyntaxError: 'return' outside function
PS C:\Users\Padmavathi> & C:/Users/Padmavathi/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/Padmavathi/
#Task 1.py"
Enter a string to reverse: engineering
Reversed string: gnireenigne
Enter a string to reverse: employee
Reversed string: eeyolpme
Enter a string to reverse: future
Reversed string: erutuf
Reversed string: erutuf
PS C:\Users\Padmavathi> 
```

□ A **built-in / slicing-based** string reversal approach



```python
#Get user input
# user_input= input("Enter a string to reverse: ")
#Call the function and display the reversed string
reversed_str = reverse_string_loop(user_input)
print(f"Reversed string: {reversed_str}")
#Write a Python program to reverse a string using slicing.
def reverse_string_slicing(s):
    return s[::-1]
user_input = input("Enter a string to reverse: ")
print(f"Reversed string: {reverse_string_slicing (user_input)}")
```

```
SyntaxError: f-string: single '}' is not allowed
PS C:\Users\Padmavathi> & C:/Users/Padmavathi/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/Padmavathi/
#Task 1.py"
Enter a string to reverse: welcome
Reversed string: emoclew
Enter a string to reverse: hello
Reversed string: olleh
Enter a string to reverse: people
Reversed string: elpoep
Reversed string: elpoep
Enter a string to reverse: meghana
Reversed string: anahgem
PS C:\Users\Padmavathi> 
```

2. Time Complexity

- **Loop-Based Approach:**

  Time complexity is **O(n)**, where *n* is the length of the string.
- **Built-in Approach:**

Time complexity is also **O(n)**, as the string must be traversed internally.

Conclusion

Although both approaches have the same time complexity, the built-in slicing method is more efficient and readable, making it the preferred choice for practical applications.