Lab Assignment 1.2 – AI Assisted Coding

N.Sai Akash

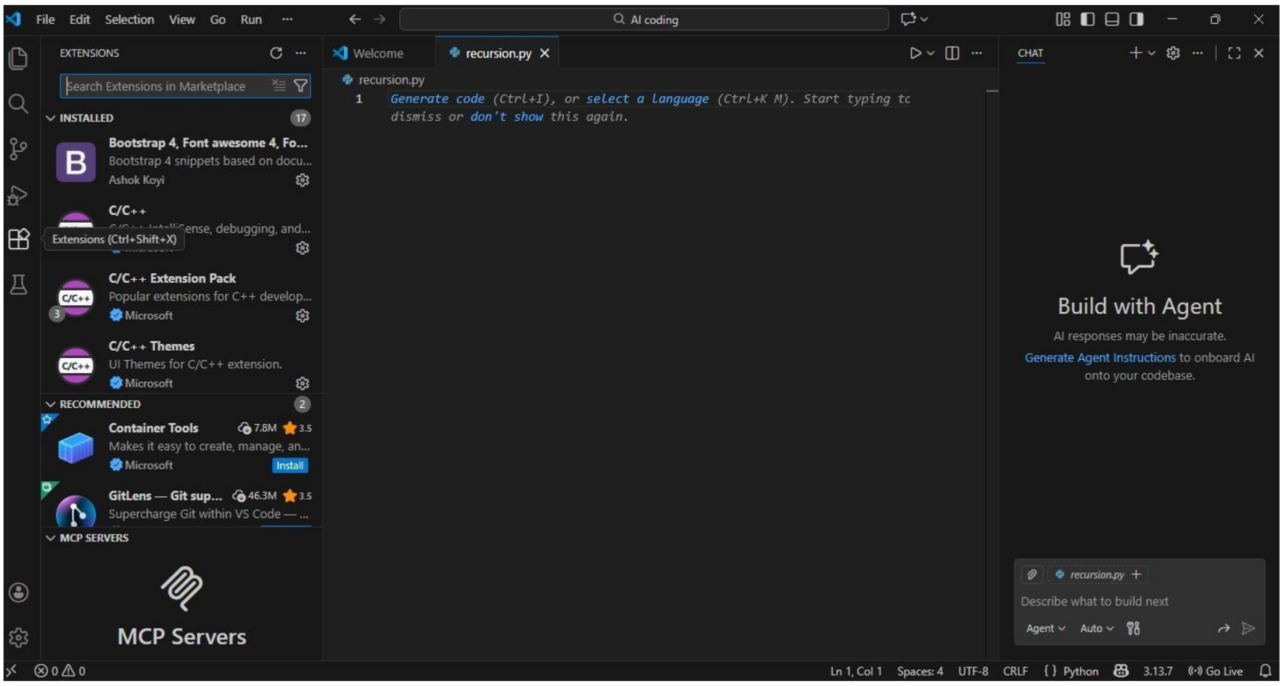
2403A51L57

B:52

Task 0: GitHub Copilot Installation & Configuration

Steps Followed:

1. Installed **Visual Studio Code**
2. Opened **Extensions Marketplace**



```
Generate code (Ctrl+I), or select a language (Ctrl+K M). Start typing to
dismiss or don't show this again.
```

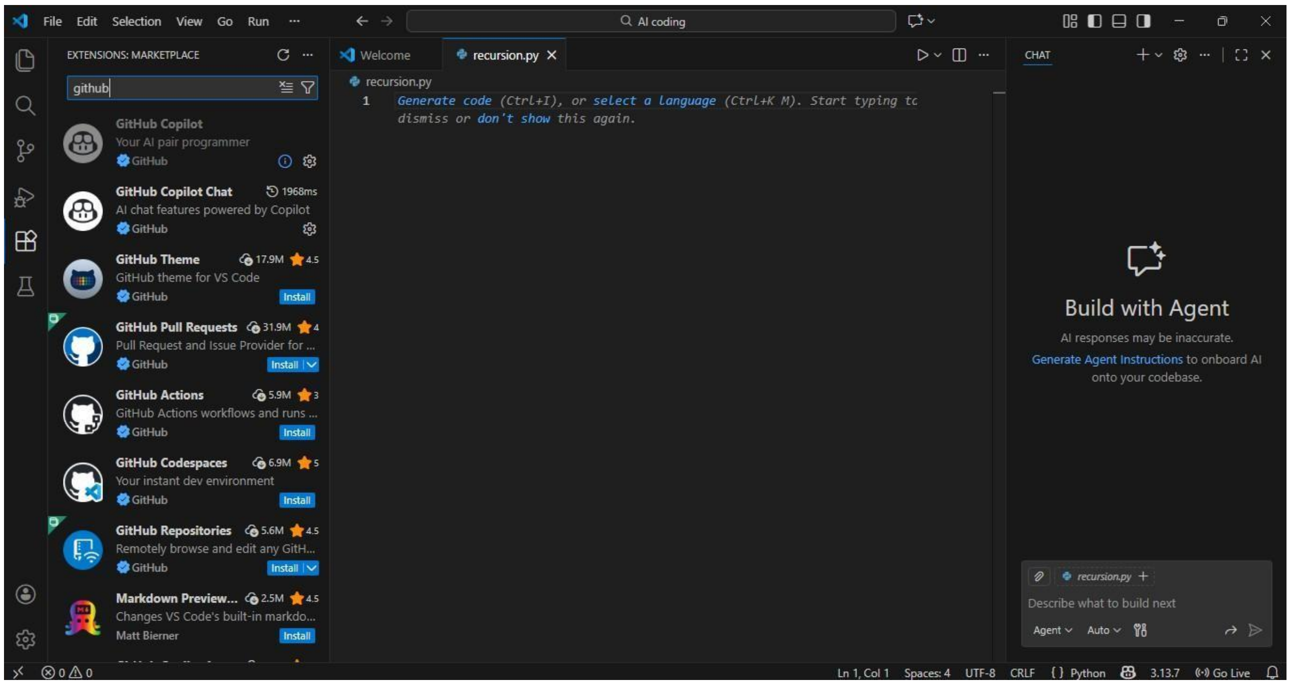
Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

File Edit Selection View Go Run
AI coding
EXTENSIONS
Search Extensions in Marketplace
INSTALLED 17
Bootstrap 4, Font awesome 4, Fo...
Bootstrap 4 snippets based on docu...
Ashok Koyi
C/C++
Extensions (Ctrl+Shift+X)
C/C++ Extension Pack
Popular extensions for C++ develop...
Microsoft
C/C++ Themes
UI Themes for C/C++ extension.
Microsoft
RECOMMENDED 2
Container Tools 7.8M 3.5
Makes it easy to create, manage, an...
Microsoft
Install
GitLens — Git sup... 46.3M 3.5
Supercharge Git within VS Code — ...
MCP SERVERS
MCP Servers
Welcome
recursion.py
CHAT
Describe what to build next
Agent
Auto
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.7 Go Live

3. Searched for **GitHub Copilot**


File Edit Selection View Go Run ...
AI coding
EXTENSIONS: MARKETPLACE
github
GitHub Copilot
Your AI pair programmer
GitHub
GitHub Copilot Chat
AI chat features powered by Copilot
GitHub Theme
GitHub theme for VS Code
GitHub Pull Requests
Pull Request and Issue Provider for ...
GitHub Actions
GitHub Actions workflows and runs ...
GitHub Codespaces
Your instant dev environment
GitHub Repositories
Remotely browse and edit any GitH...
Markdown Preview...
Changes VS Code's built-in markdo...
Matt Bierner
Install
Welcome
recursion.py
CHAT
Build with Agent
Agent
Auto
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.7 Go Live

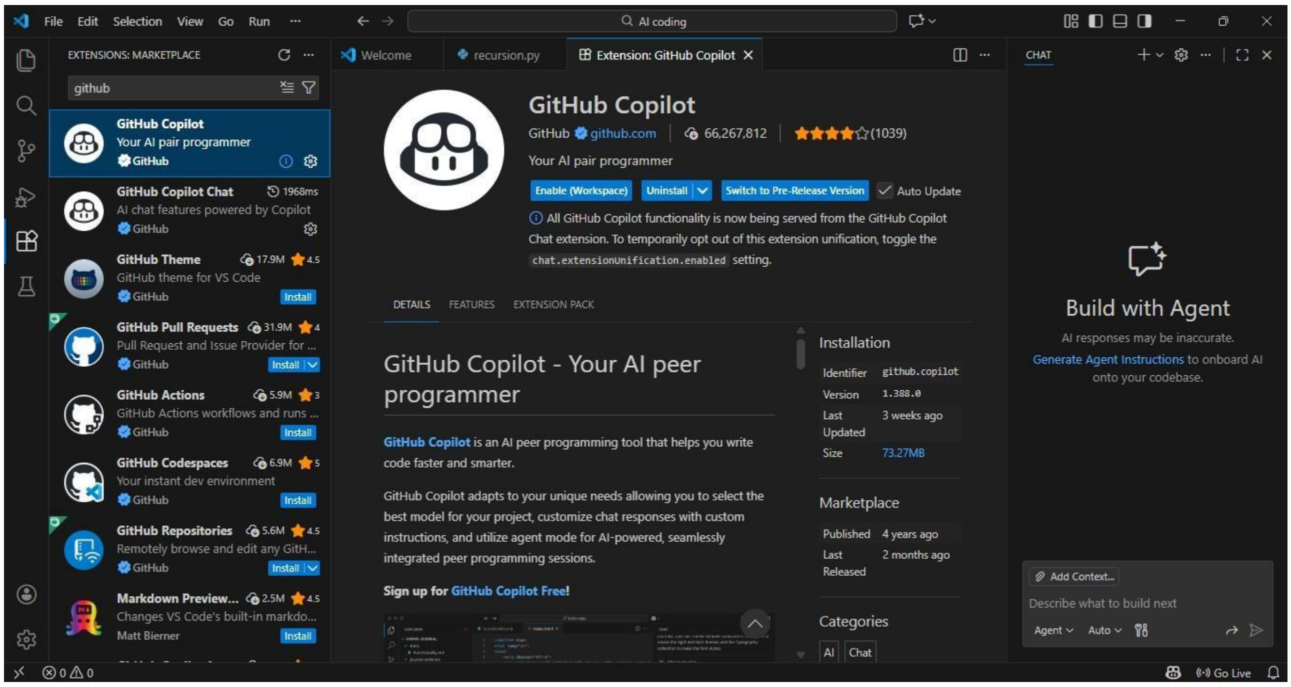
```
Generate code (Ctrl+I), or select a language (Ctrl+K M). Start typing tc
dismiss or don't show this again.
```

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next

4. Clicked **Install**



GitHub Copilot

GitHub github.com | 66,267,812 | ★★★★☆(1039)

Your AI pair programmer

All GitHub Copilot functionality is now being served from the GitHub Copilot Chat extension. To temporarily opt out of this extension unification, toggle the `chat.extensionUnification.enabled` setting.

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!

Installation

Identifier	github.copilot
Version	1.388.0
Last Updated	3 weeks ago
Size	73.27MB

Marketplace

Published	4 years ago
Last Released	2 months ago

Categories

AI Chat

AI responses may be inaccurate.

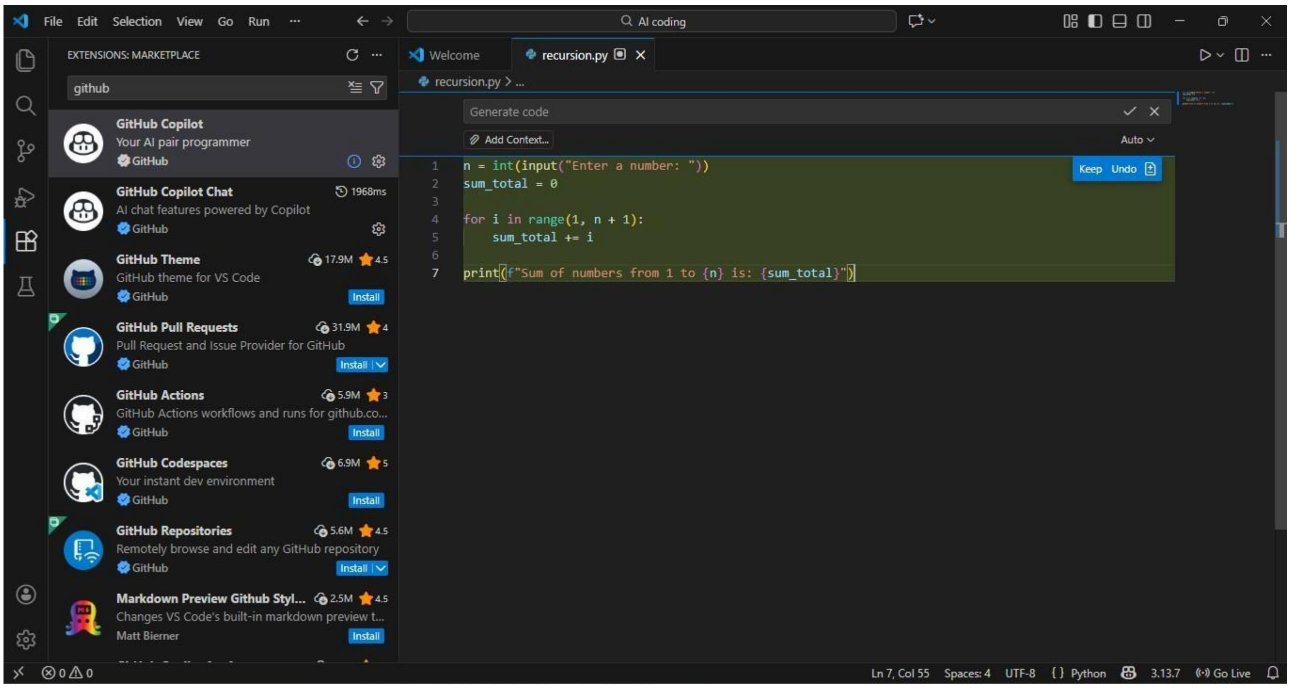
Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next

5. Signed in with **GitHub Account**

6. Enabled Copilot suggestions

7. Verified Copilot inline suggestions in Python file

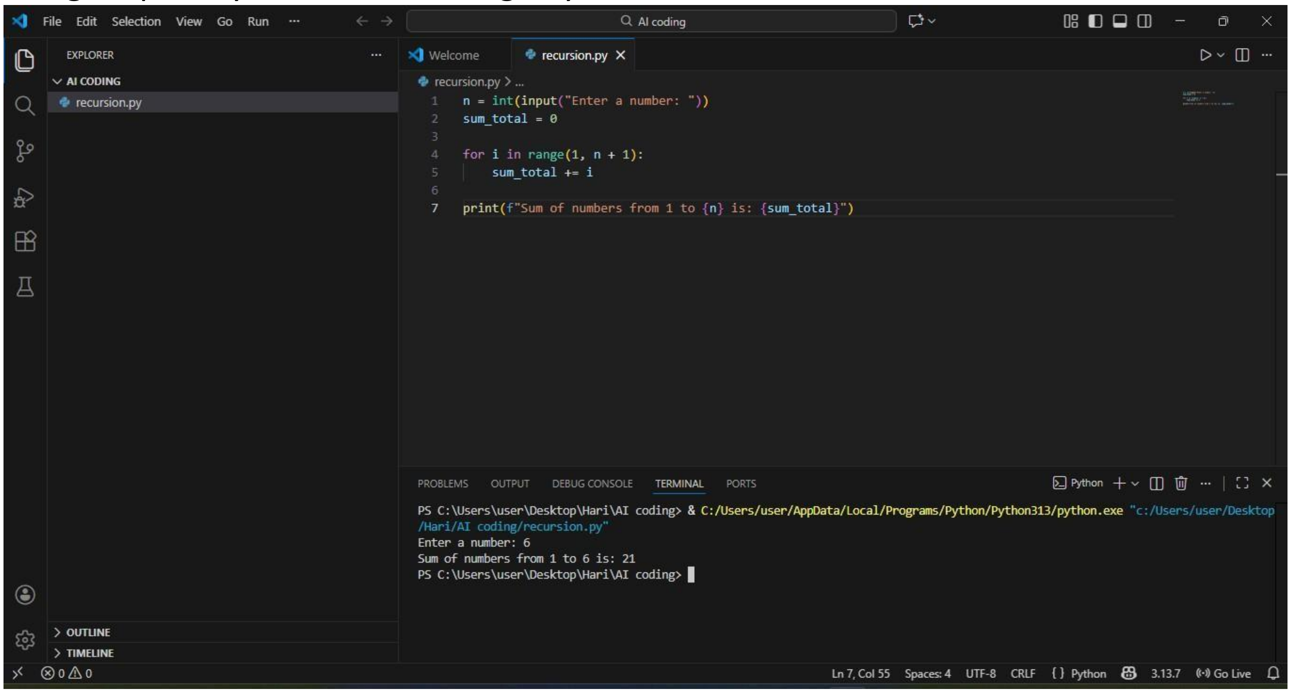

EXTENSIONS: MARKETPLACE
github
GitHub Copilot
Your AI pair programmer
GitHub Copilot Chat
AI chat features powered by Copilot
GitHub Theme
GitHub theme for VS Code
GitHub Pull Requests
Pull Request and Issue Provider for GitHub
GitHub Actions
GitHub Actions workflows and runs for github.co...
GitHub Codespaces
Your instant dev environment
GitHub Repositories
Remotely browse and edit any GitHub repository
Markdown Preview Github Styl...
Changes VS Code's built-in markdown preview t...
Generate code
Add Context...
Keep
Undo

```python
n = int(input("Enter a number: "))
sum_total = 0

for i in range(1, n + 1):
    sum_total += i

print(f"Sum of numbers from 1 to {n} is: {sum_total}")
```

Task 1: AI-Generated Logic Without Modularization

(Factorial without Functions)

Prompt Used: "Write a Python program to calculate factorial of a number using loops only, without defining any function."



```python
n = int(input("Enter a number: "))
sum_total = 0

for i in range(1, n + 1):
    sum_total += i

print(f"Sum of numbers from 1 to {n} is: {sum_total}")
```

```
PS C:\Users\user\Desktop\Hari\AI coding> & C:/Users/user/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/user/Desktop/Hari/AI coding/recursion.py"
Enter a number: 6
Sum of numbers from 1 to 6 is: 21
PS C:\Users\user\Desktop\Hari\AI coding>
```

GitHub Copilot was very helpful for a beginner as it generated correct logic instantly.

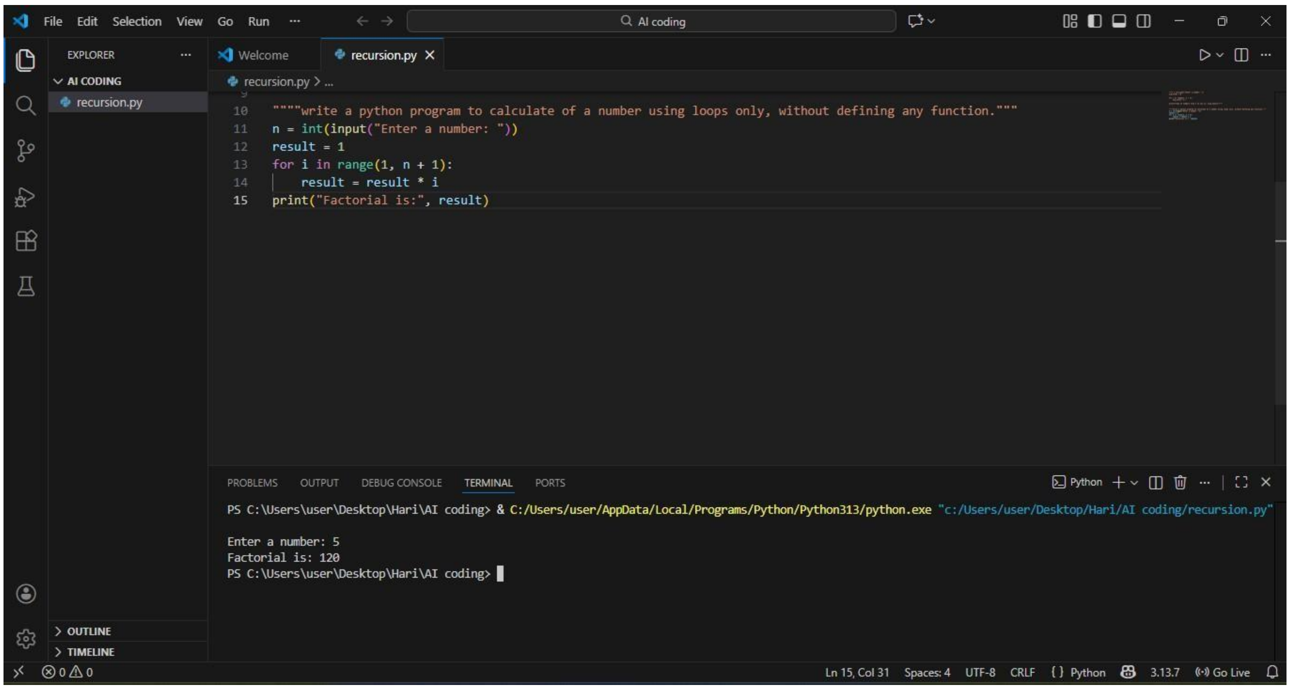
It followed basic Python syntax and loop structure accurately.

The code was readable and easy to understand.

However, it did not include input validation automatically.
Best practices like modular design were not applied unless explicitly prompted.

Task 2: AI Code Optimization & Cleanup

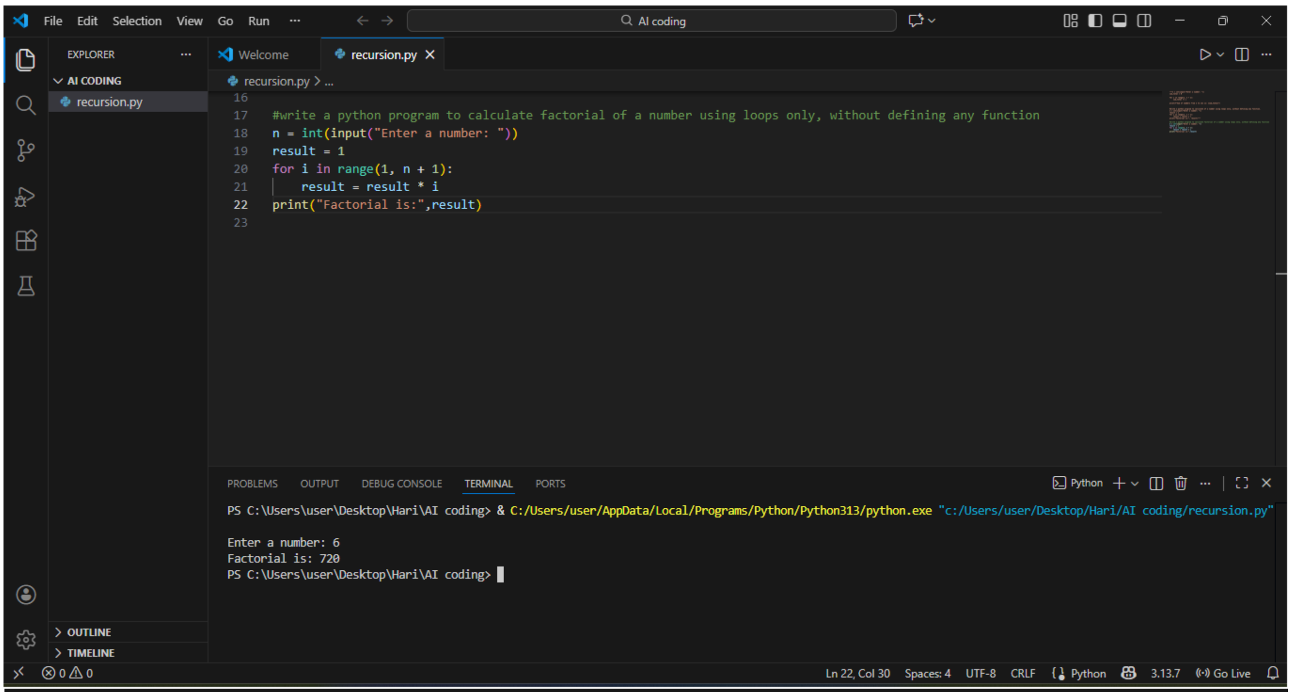
Original Code:



```python
""""write a python program to calculate of a number using loops only, without defining any function."""
n = int(input("Enter a number: "))
result = 1
for i in range(1, n + 1):
    result = result * i
print("Factorial is:", result)
```

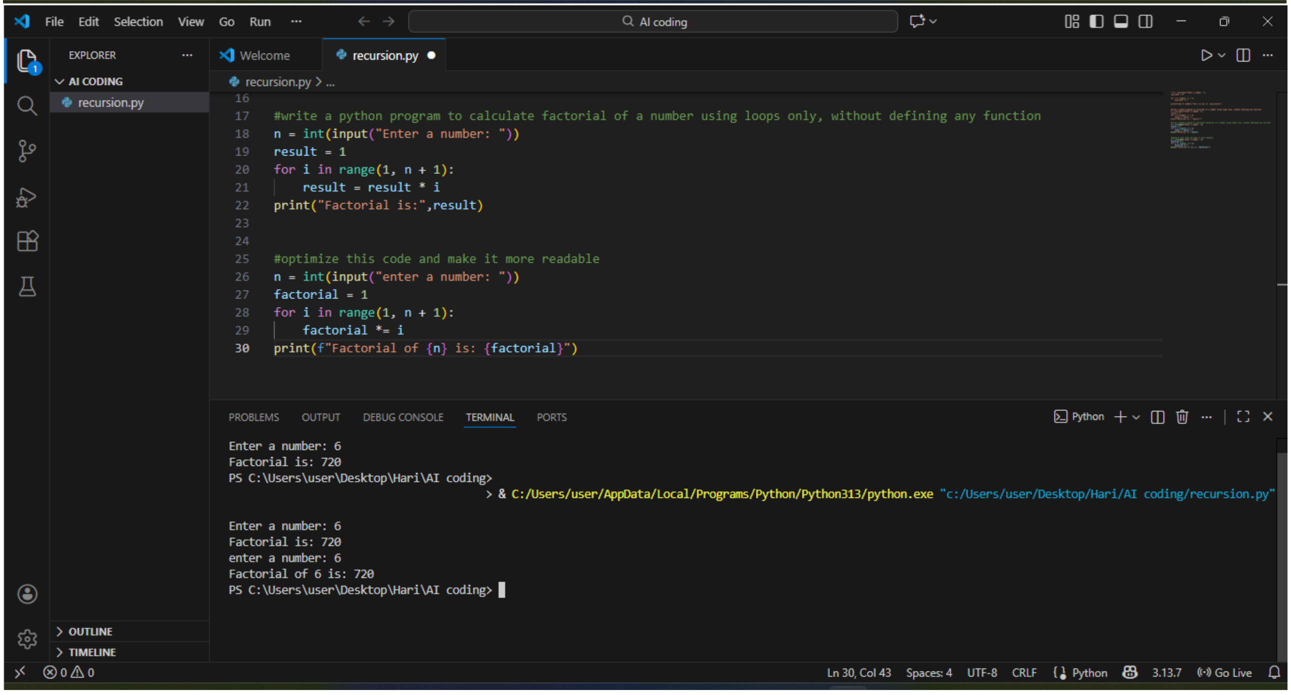
```
PS C:\Users\user\Desktop\Hari\AI coding> & C:/Users/user/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/user/Desktop/Hari/AI coding/recursion.py"

Enter a number: 5
Factorial is: 120
PS C:\Users\user\Desktop\Hari\AI coding>
```

Prompt Used: “Optimize this code and make it more readable”



```python
#write a python program to calculate factorial of a number using loops only, without defining any function
n = int(input("Enter a number: "))
result = 1
for i in range(1, n + 1):
    result = result * i
print("Factorial is:",result)
```

```
PS C:\Users\user\Desktop\Hari\AI coding> & C:/Users/user/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/user/Desktop/Hari/AI coding/recursion.py"

Enter a number: 6
Factorial is: 720
PS C:\Users\user\Desktop\Hari\AI coding>
```



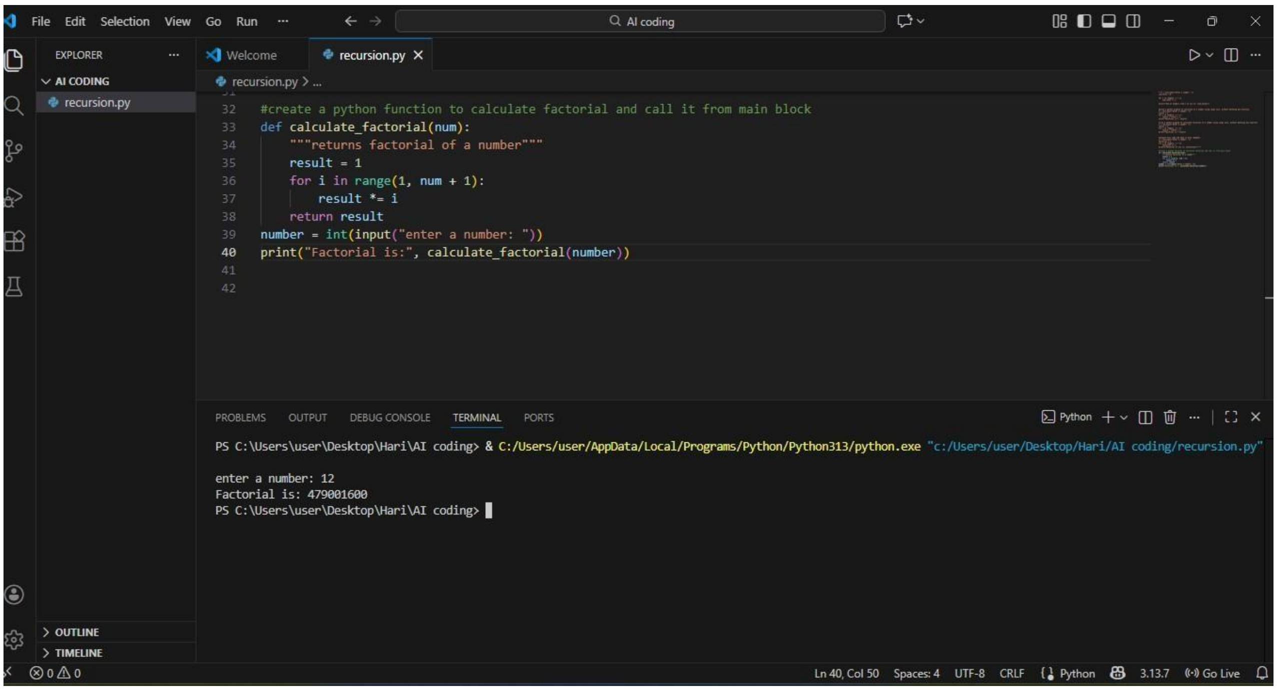
```python
#write a python program to calculate factorial of a number using loops only, without defining any function
n = int(input("Enter a number: "))
result = 1
for i in range(1, n + 1):
    result = result * i
print("Factorial is:",result)


#optimize this code and make it more readable
n = int(input("enter a number: "))
factorial = 1
for i in range(1, n + 1):
    factorial *= i
print(f"Factorial of {n} is: {factorial}")
```

```
Enter a number: 6
Factorial is: 720
PS C:\Users\user\Desktop\Hari\AI coding>
                                         > & C:/Users/user/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/user/Desktop/Hari/AI coding/recursion.py"

Enter a number: 6
Factorial is: 720
enter a number: 6
Factorial of 6 is: 720
PS C:\Users\user\Desktop\Hari\AI coding>
```

The optimized version improves clarity, maintainability, and readability without affecting performance.

Task 3: Modular Design Using AI Assistance (Factorial with Functions)

Prompt Used: “Create a Python function to calculate factorial and call it from main block”



```python
#create a python function to calculate factorial and call it from main block
def calculate_factorial(num):
    """returns factorial of a number"""
    result = 1
    for i in range(1, num + 1):
        result *= i
    return result
number = int(input("enter a number: "))
print("Factorial is:", calculate_factorial(number))
```

```
PS C:\Users\user\Desktop\Hari\AI coding> & C:/Users/user/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/user/Desktop/Hari/AI coding/recursion.py"

enter a number: 12
Factorial is: 479001600
PS C:\Users\user\Desktop\Hari\AI coding>
```

Modularity improves reusability by allowing the same function to be used across multiple programs. It also simplifies testing and debugging.

Task 4: Comparative Analysis

Procedural vs Modular AI Code

Criteria	**Without Function**	**With Function**
Logic Clarity	Moderate	High
Reusability	No	Yes

Debugging Ease	Difficult	Easy
Large Project Suitability	Poor	Excellent
AI Dependency Risk	Higher	Lower

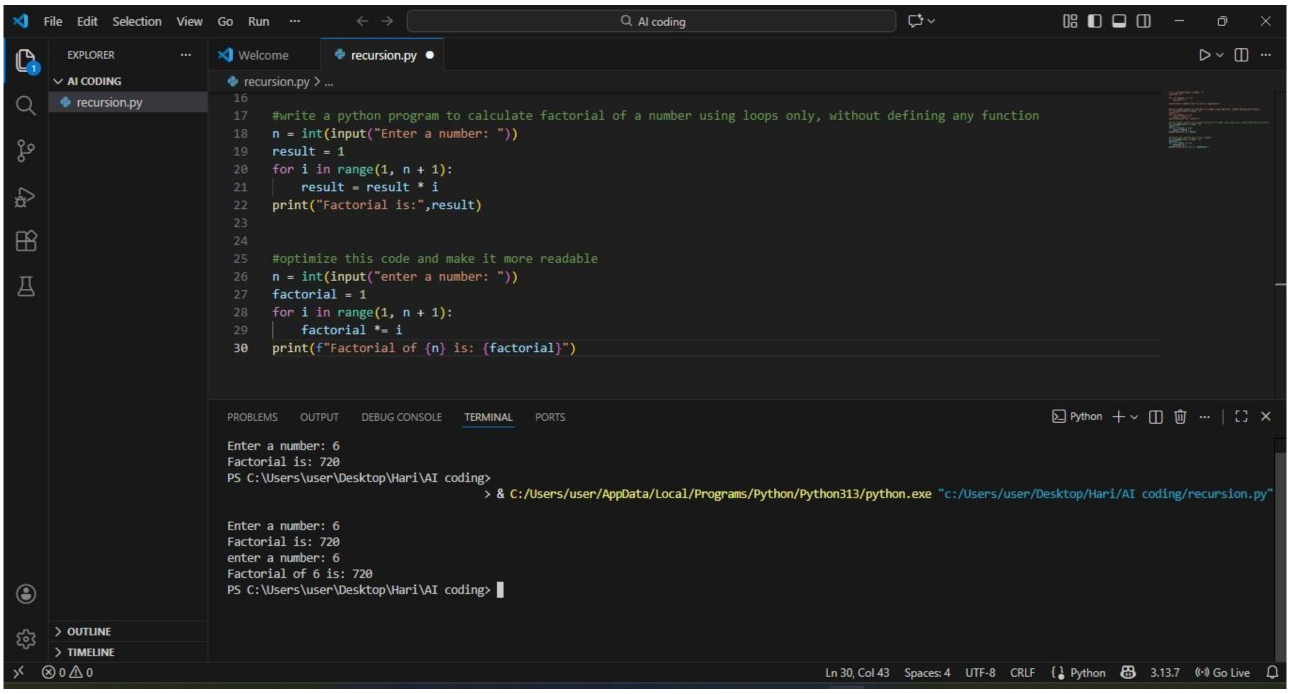
Conclusion:

Function-based design is more scalable and suitable for real-world applications.

Task 5: Iterative vs Recursive AI Code

Prompt Used: "Generate iterative and recursive factorial programs in Python"

Execution Flow Explanation:



```python

#write a python program to calculate factorial of a number using loops only, without defining any function
n = int(input("Enter a number: "))
result = 1
for i in range(1, n + 1):
    result = result * i
print("Factorial is:",result)


#optimize this code and make it more readable
n = int(input("enter a number: "))
factorial = 1
for i in range(1, n + 1):
    factorial *= i
print(f"Factorial of {n} is: {factorial}")
```

```
Enter a number: 6
Factorial is: 720
PS C:\Users\user\Desktop\Hari\AI coding>
                                         > & C:/Users/user/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/user/Desktop/Hari/AI coding/recursion.py"

Enter a number: 6
Factorial is: 720
enter a number: 6
Factorial of 6 is: 720
PS C:\Users\user\Desktop\Hari\AI coding>
```

- Iterative version uses a loop and constant memory.
- Recursive version uses function calls and stack memory.

Comparison:

Aspect	Iterative	Recursive
Readability	Simple	Elegant

Stack Usage	No	Yes
Performance	Faster	Slower
Risk	Low	Stack Overflow
Recommendation	Preferred	Avoid for large inputs