Assignment : 1.2

Name :Saisharan

Ht.no :2303A51434

Course Name: Ai Assistant Coding Batch:21

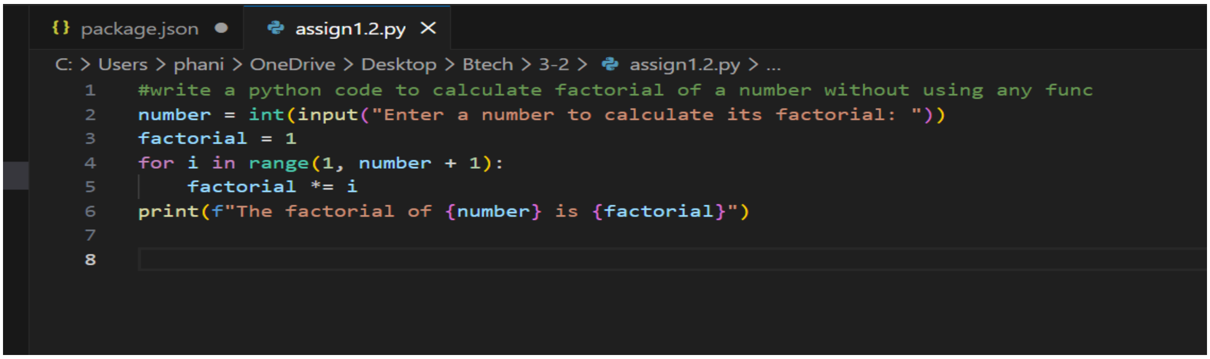
Task 1: AI-Generated Logic Without Modularization (Factorial without Functions)

- Scenario

You are building a small command-line utility for a startup intern onboarding task. The program is simple and must be written quickly without modular design.

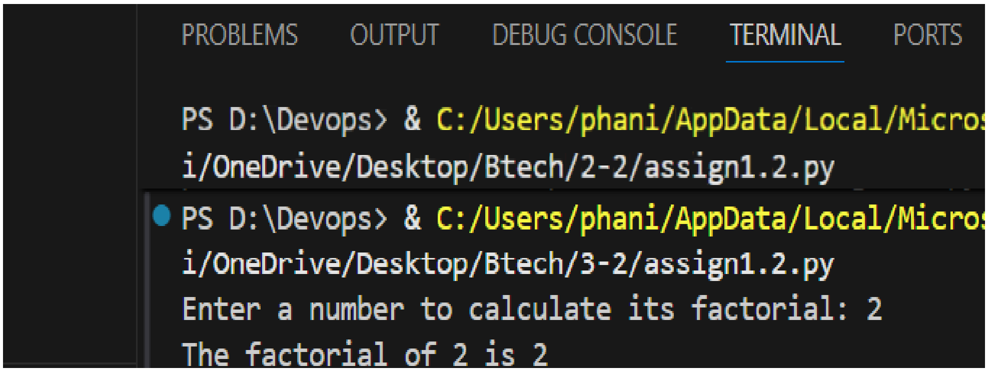
- Task Description

Use GitHub Copilot to generate a Python program that computes a mathematical product-based value (factorial-like logic) directly in the main execution flow, without using any user-defined functions


{} package.json
assign1.2.py
C: > Users > phani > OneDrive > Desktop > Btech > 3-2 > assign1.2.py > ...

```python
#write a python code to calculate factorial of a number without using any func
number = int(input("Enter a number to calculate its factorial: "))
factorial = 1
for i in range(1, number + 1):
    factorial *= i
print(f"The factorial of {number} is {factorial}")
```

Output:



```
PS D:\Devops> & C:/Users/phani/AppData/Local/Micros
i/OneDrive/Desktop/Btech/2-2/assign1.2.py
PS D:\Devops> & C:/Users/phani/AppData/Local/Micros
i/OneDrive/Desktop/Btech/3-2/assign1.2.py
Enter a number to calculate its factorial: 2
The factorial of 2 is 2
```

Task 2: AI Code Optimization & Cleanup (Improving Efficiency)

- Scenario

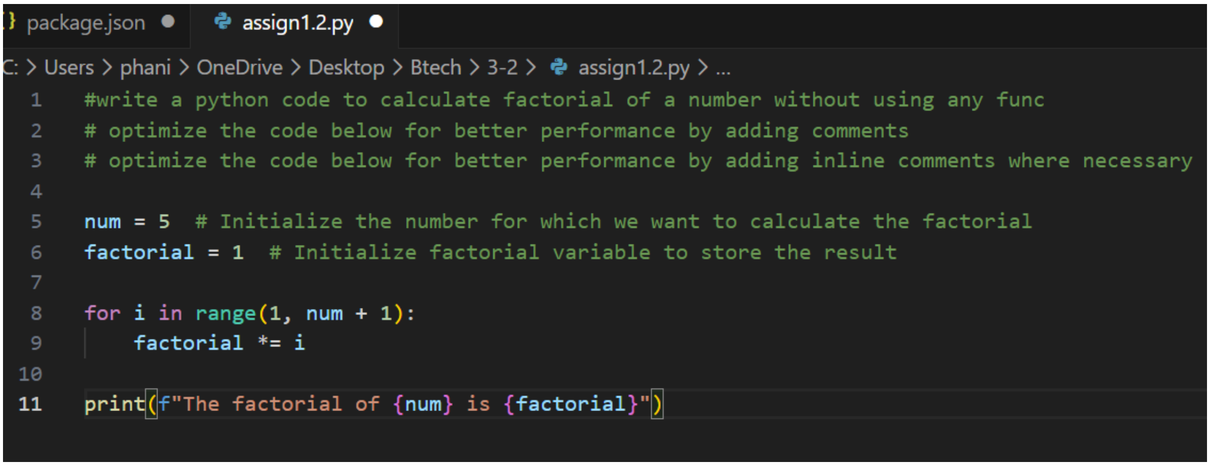
Your team lead asks you to review AI-generated code before committing it to a shared repository.

- Task Description: Analyze the code generated in Task 1 and use Copilot again to:

➢ Reduce unnecessary variables

➢ Improve loop clarity

➢ Enhance readability and efficiency


package.json
assign1.2.py
C: > Users > phani > OneDrive > Desktop > Btech > 3-2 > assign1.2.py > ...

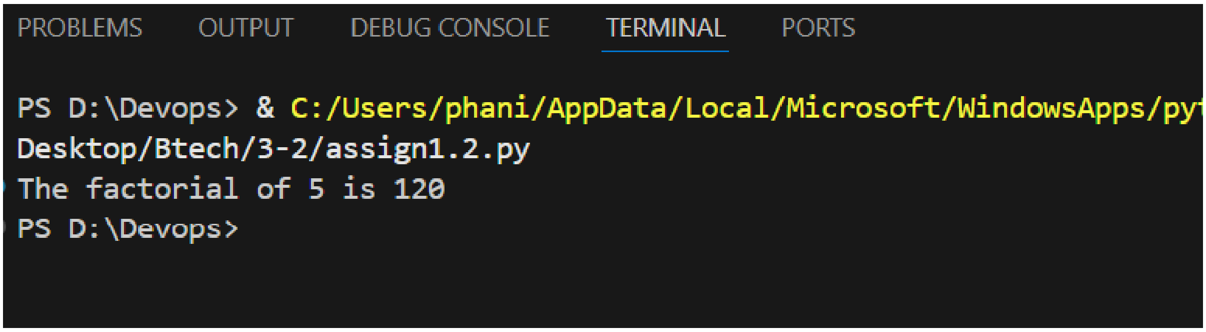
```python
#write a python code to calculate factorial of a number without using any func
# optimize the code below for better performance by adding comments
# optimize the code below for better performance by adding inline comments where necessary

num = 5  # Initialize the number for which we want to calculate the factorial
factorial = 1  # Initialize factorial variable to store the result

for i in range(1, num + 1):
    factorial *= i

print(f"The factorial of {num} is {factorial}")
```

Output:



```
PS D:\Devops> & C:/Users/phani/AppData/Local/Microsoft/WindowsApps/pyt
Desktop/Btech/3-2/assign1.2.py
The factorial of 5 is 120
PS D:\Devops>
```

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

- Scenario:

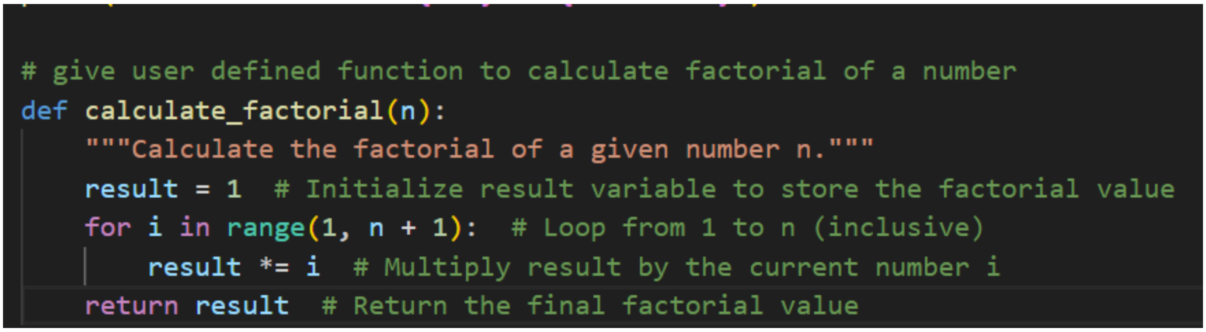
The same logic now needs to be reused in multiple scripts.

- Task Description:

Use GitHub Copilot to generate a modular version of the program by:

➢ Creating a user-defined function

➢ Calling the function from the main block



```python
# give user defined function to calculate factorial of a number
def calculate_factorial(n):
    """Calculate the factorial of a given number n."""
    result = 1  # Initialize result variable to store the factorial value
    for i in range(1, n + 1):  # Loop from 1 to n (inclusive)
        result *= i  # Multiply result by the current number i
    return result  # Return the final factorial value
```

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

- Scenario

As part of a code review meeting, you are asked to justify design choices.

- Task Description

Compare the non-function and function-based Copilot-generated programs on the following criteria:

➢ Logic clarity

➢ Reusability

➢ Debugging ease

➢ Suitability for large projects

➢ AI dependency risk