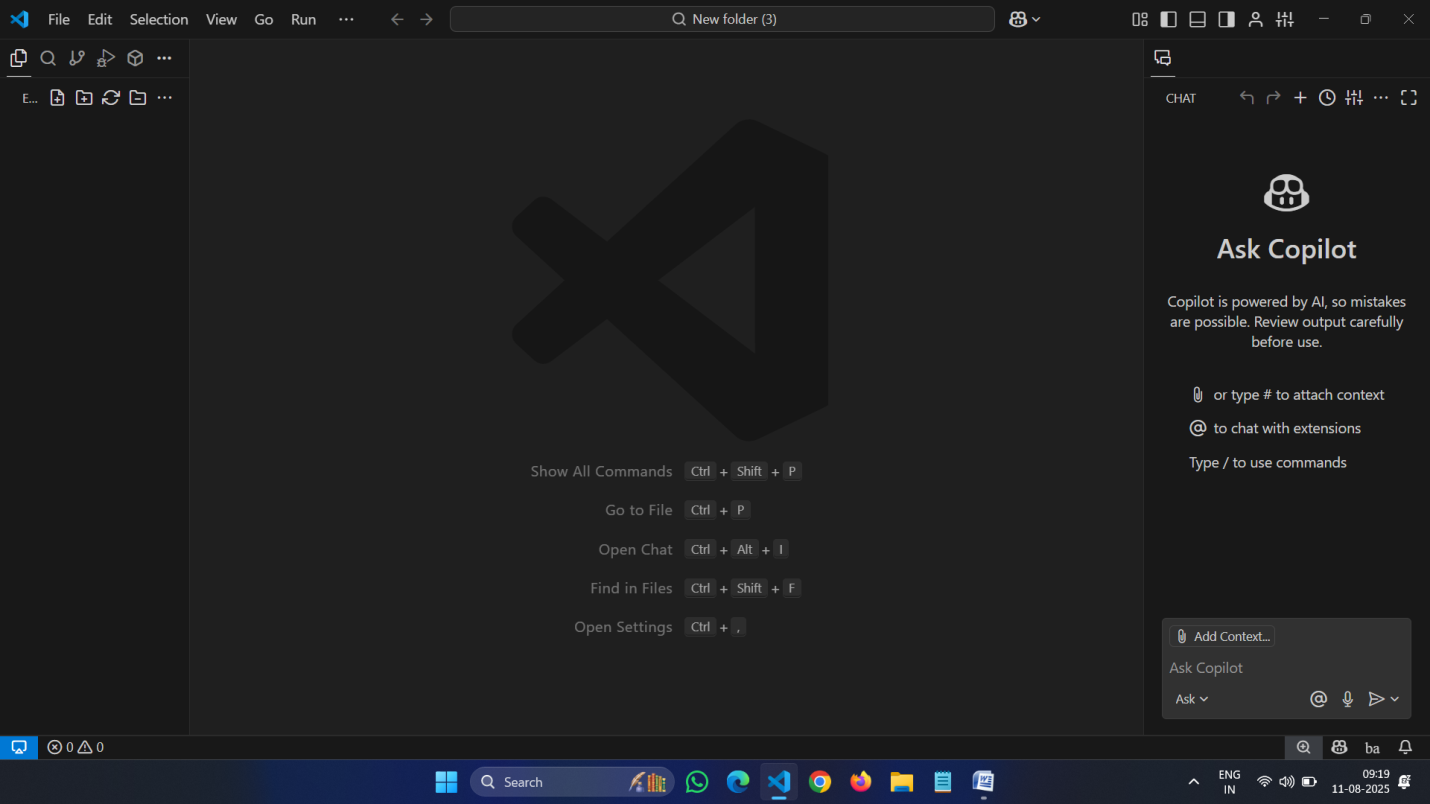
**Task 0:**

Install and configure GitHub Copilot in VS code.

Take Screenshot of each step:

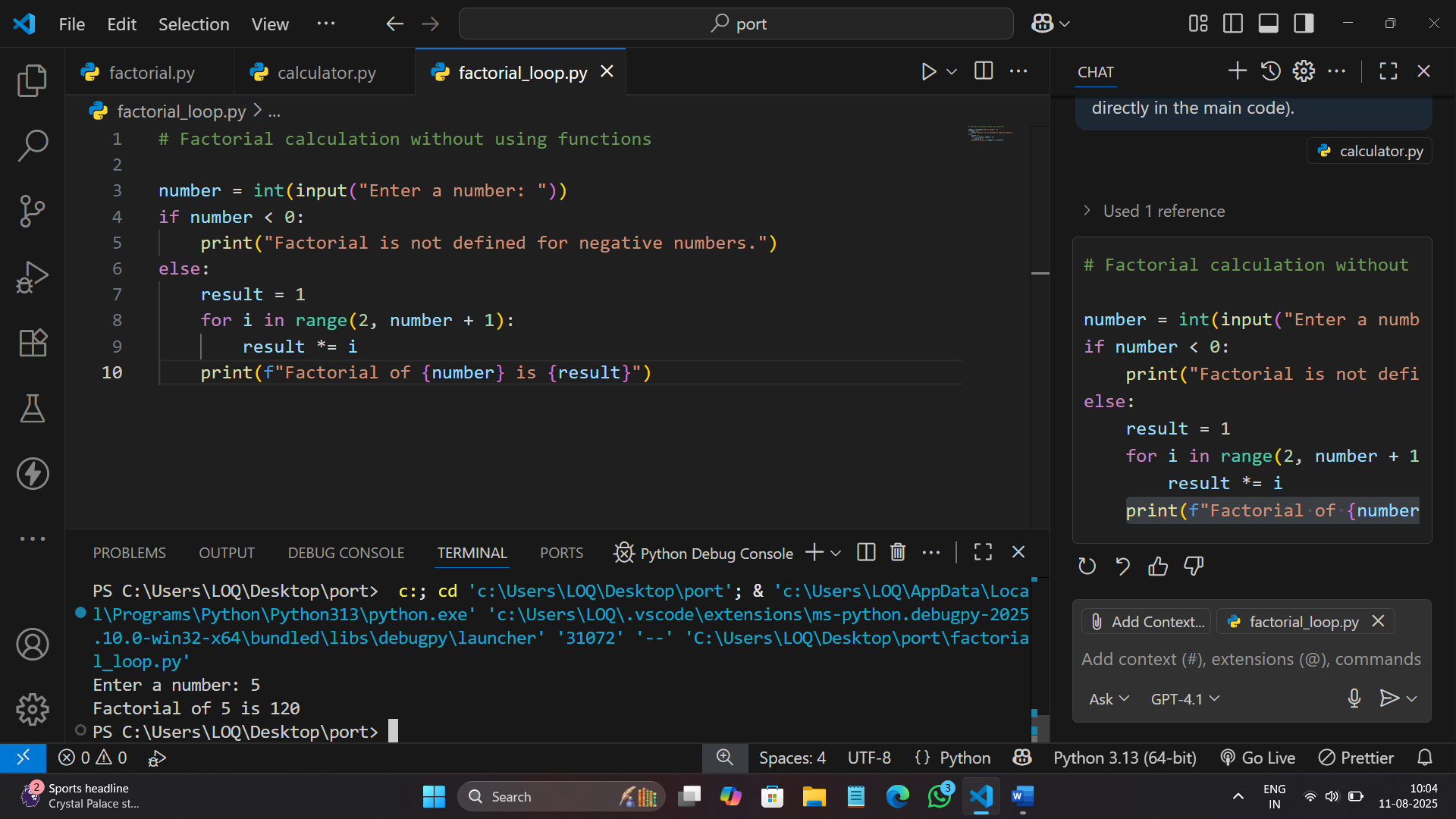


**Task 1:** Factorial without Functions

* **Description:**

Use GitHub copilot to generate python program that calculates the factorial of a number without defining any function (using loops directly in the main code).

* **Expected output:**

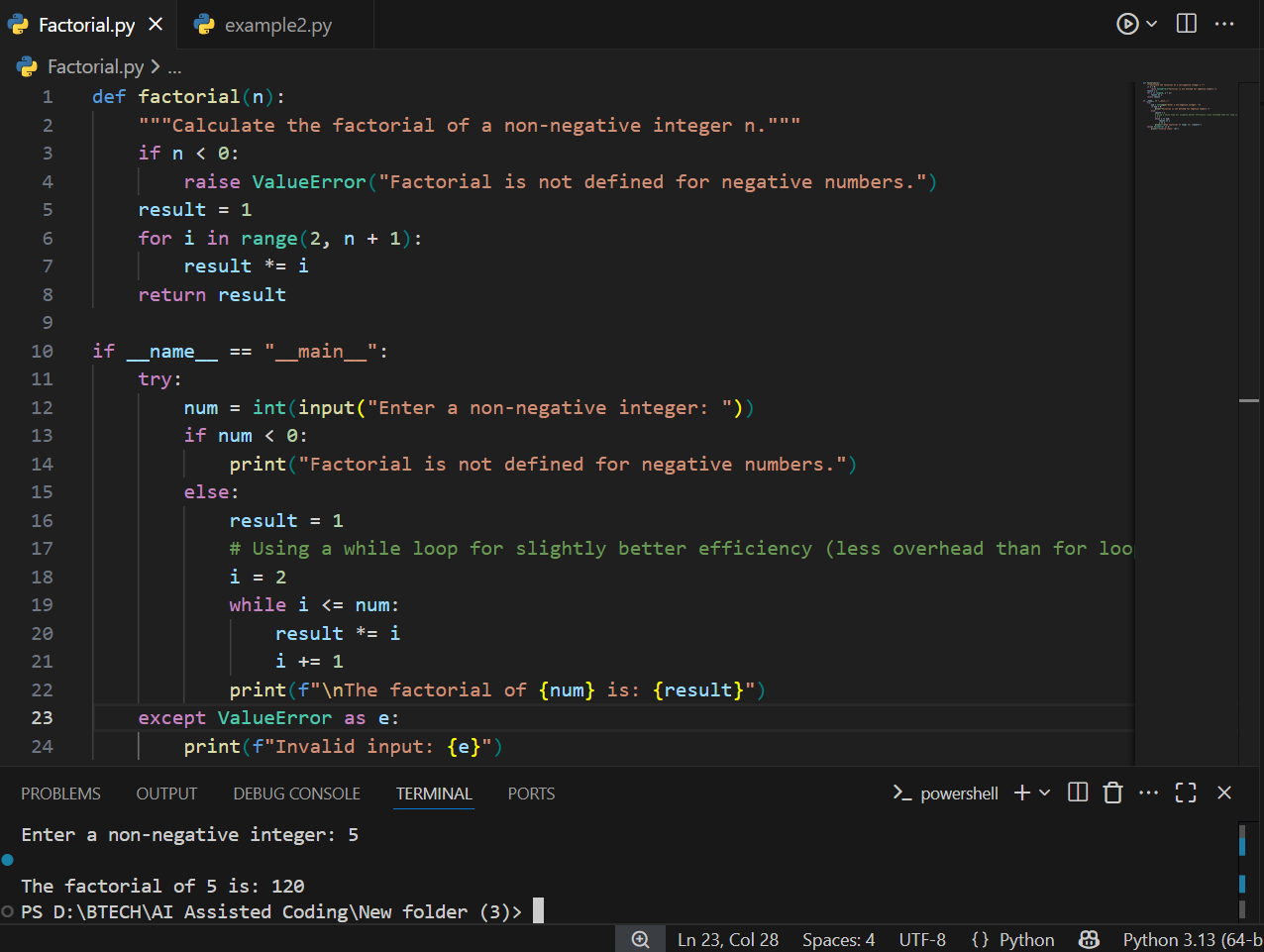


**Task 2:** Improving Efficiency

* **Description**

Examine the Copilot generated code.

The code is improved by removing the function definition and calculating the factorial directly in the main block using a loop. This reduces function call overhead and simplifies the logic. The while loop starts from 2 and multiplies up to the input number, which is efficient for factorial calculation. Error handling for invalid input is also included.

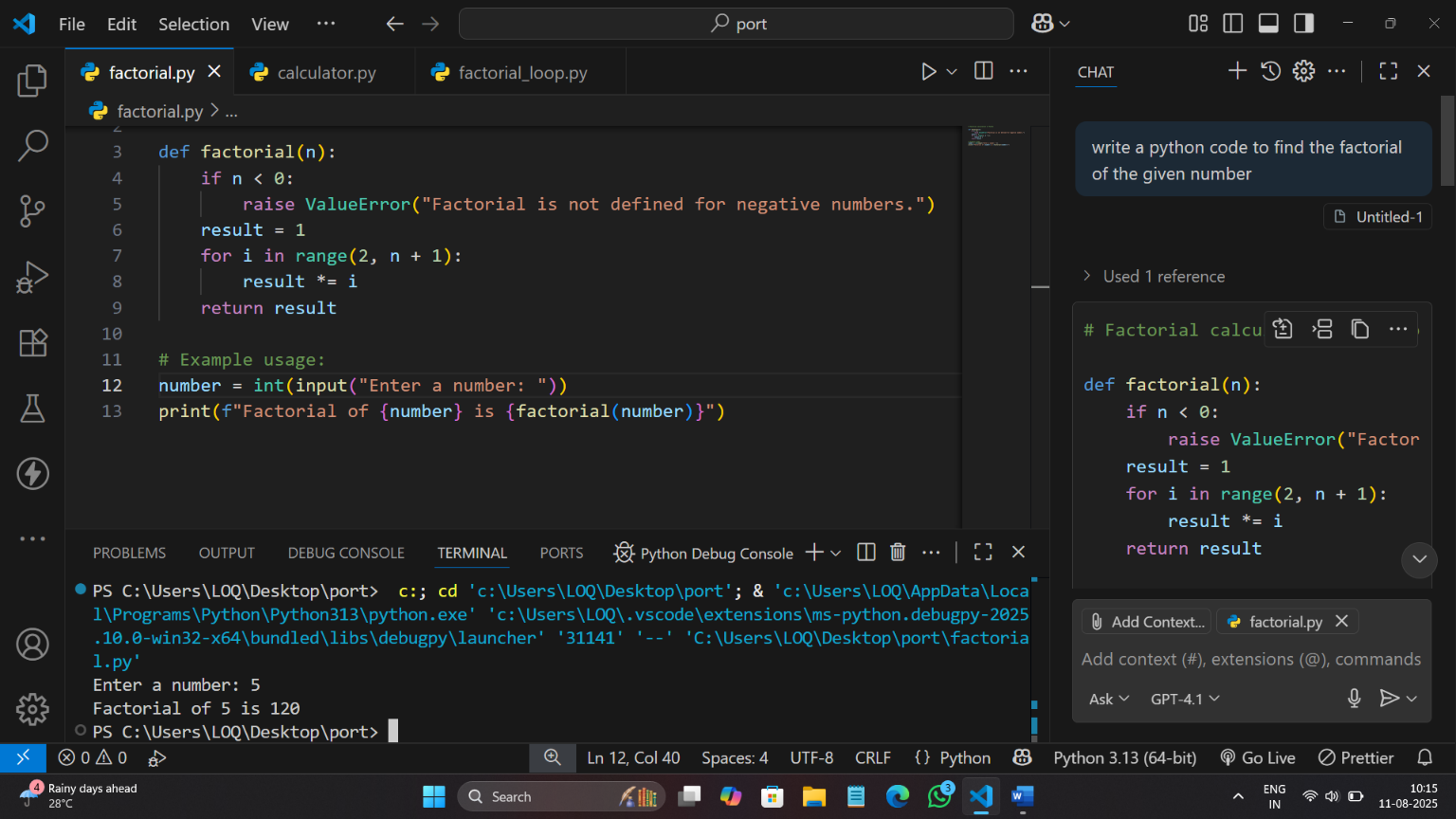


**Task 3:** Factorial with Functions

* **Description:**

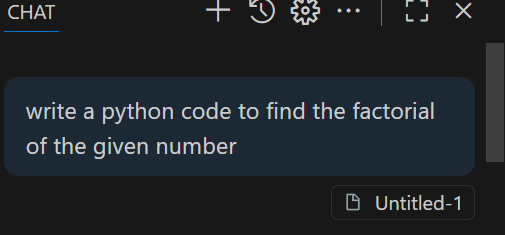
Use GitHub copilot to generate python program that calculates the factorial of a number without defining any function.

* **Expected output:**



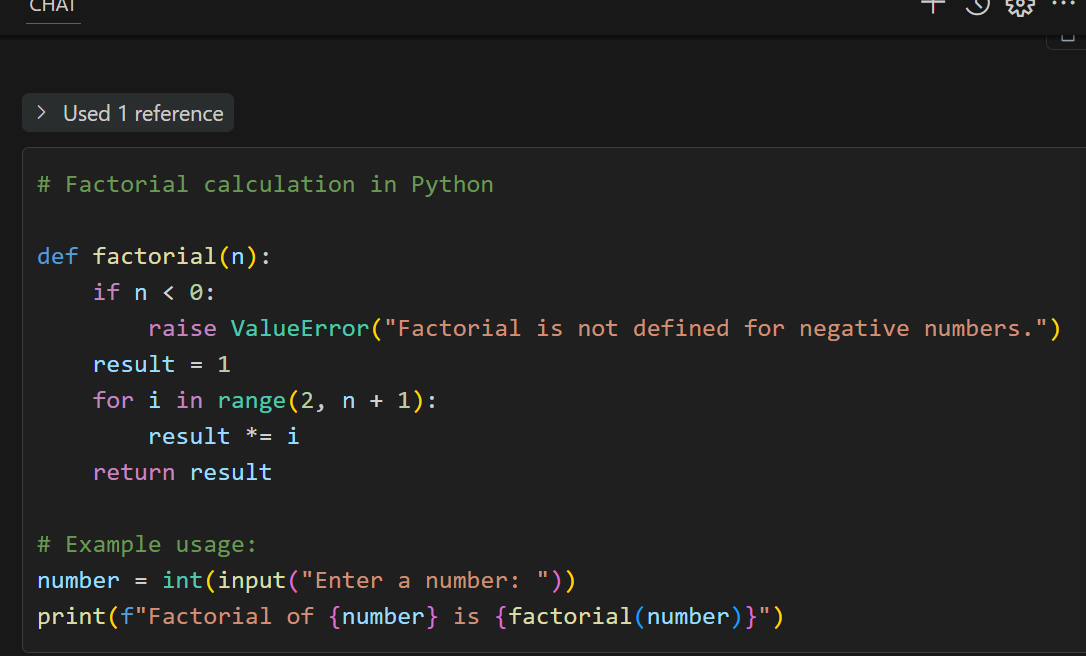
* **Step 1:**

Enter the prompt in the GitHub copilot



* **Step 2:**

Paste the code in the work space.



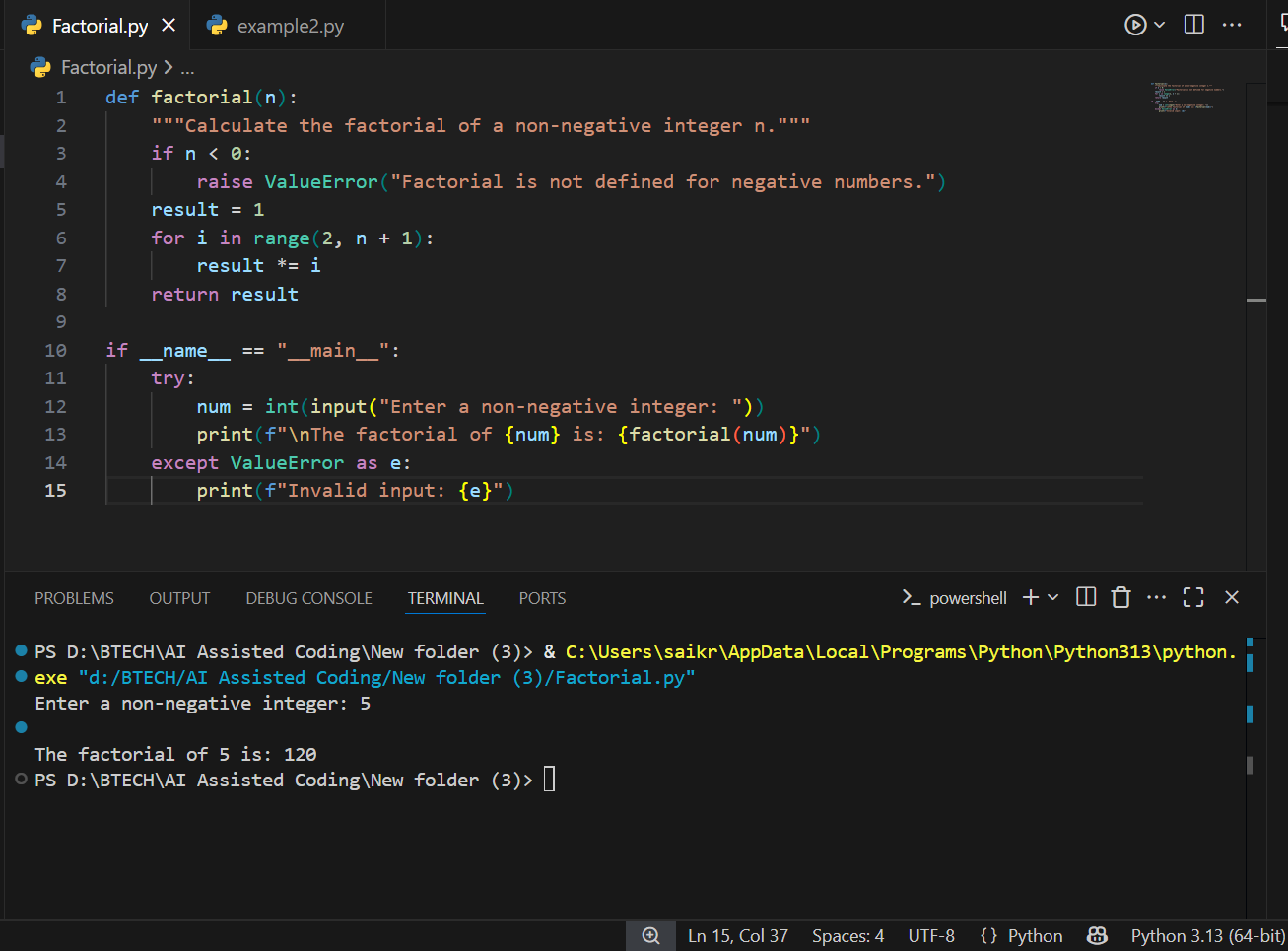
**Task 4:** Comparative Analysis – With vs Without Functions

* **Description:**

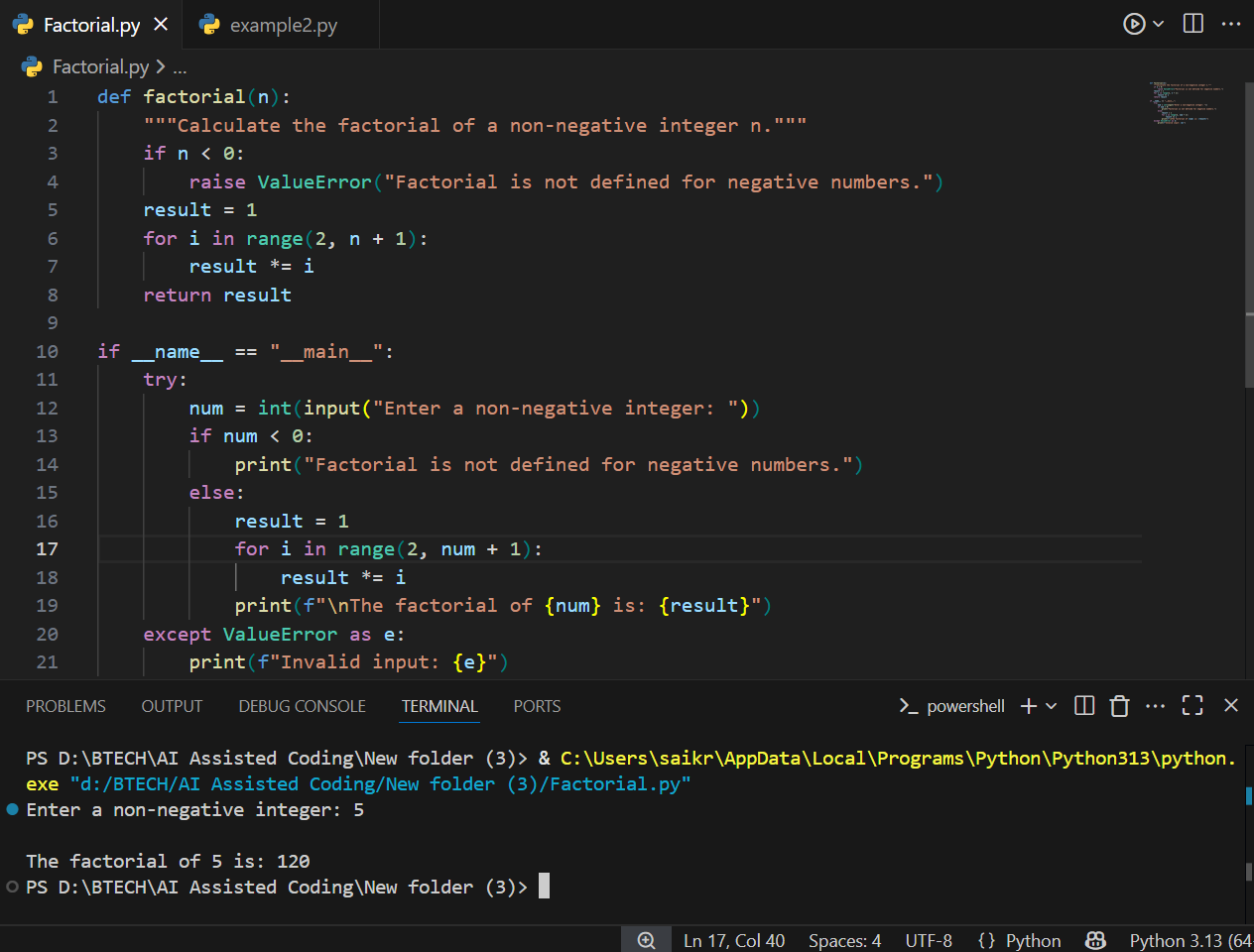
Differentiate between the copilot generated factorial program with functions and without functions in the terms of logic, reusability, and execution.

* **Expected Output:**

1. With functions



1. Without functions



**Logic**

* **With Function:**  
  The logic is encapsulated in a user-defined function ([factorial(n)](vscode-file://vscode-app/c:/Users/saikr/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)), which takes an argument and returns the factorial. The main code calls this function.
* **Without Function:**  
  The logic is written directly in the main code block. The loop and calculation are performed inline, without any encapsulation.

**2. Reusability**

* **With Function:**  
  Highly reusable. The [factorial](vscode-file://vscode-app/c:/Users/saikr/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) function can be called multiple times with different arguments, in other parts of the program or from other modules.
* **Without Function:**  
  Not reusable. The code can only be executed as written, and cannot be called with different inputs without rewriting or copying the logic.

**3. Execution**

* **With Function:**  
  Slight overhead due to function calls, but allows for cleaner code, easier testing, and better error handling.
* **Without Function:**  
  Executes sequentially in the main block. Slightly faster for very simple scripts due to no function call overhead, but less organized and harder to maintain for larger programs.

**Summary:**  
Using functions improves code organization, reusability, and maintainability,

while direct logic in the main block is simpler but less flexible and harder to reuse.

**Task 5:** Iterative vs Recursive Factorial

* **Description:**

Prompt GitHub Copilot to generate both iterative and recursive versions of the factorial function

* **Expected Output:**

