**ASSIGNMENT – 1**

* **Installing GitHub Copilot in VS Code:**

A screenshot of a computer

AI-generated content may be incorrect.

**TASK-1:** Generate a Python program that calculates the factorial of a number without defining any functions.

**Code:**

A screenshot of a computer

AI-generated content may be incorrect.

**Output:**

**A black screen with colorful text

AI-generated content may be incorrect.**

**Explanation:**

This program calculates the factorial of a given number entered by the user.

**Step-by-step explanation:**

1. **Input:**  
   The program prompts the user to enter a number and converts it to an integer with [int(input("Enter a number: "))](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
2. **Initialization:**  
   It initializes a variable [factorial](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) to 1. This will be used to store the result.
3. **Negative Number Check:**  
   If the number is less than 0, it prints a message saying factorial does not exist for negative numbers.
4. **Zero Check:**  
   If the number is 0, it prints that the factorial of 0 is 1 (by definition).
5. **Factorial Calculation:**  
   If the number is positive, it uses a for loop from 1 to [num](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) (inclusive), multiplying [factorial](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) by each value of [i](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) in the loop.
6. **Output:**  
   Finally, it prints the factorial of the entered number.

**Example:**  
If the user enters 5, the output will be:  
The factorial of 5 is 120.

**TASK-2:** **-**Generate code from Task 1 and demonstrate how its efficiency can be improved (e.g., removing unnecessary variables, optimizing loops).

**Code:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Output:**

**A black background with many small colored lines

AI-generated content may be incorrect.**

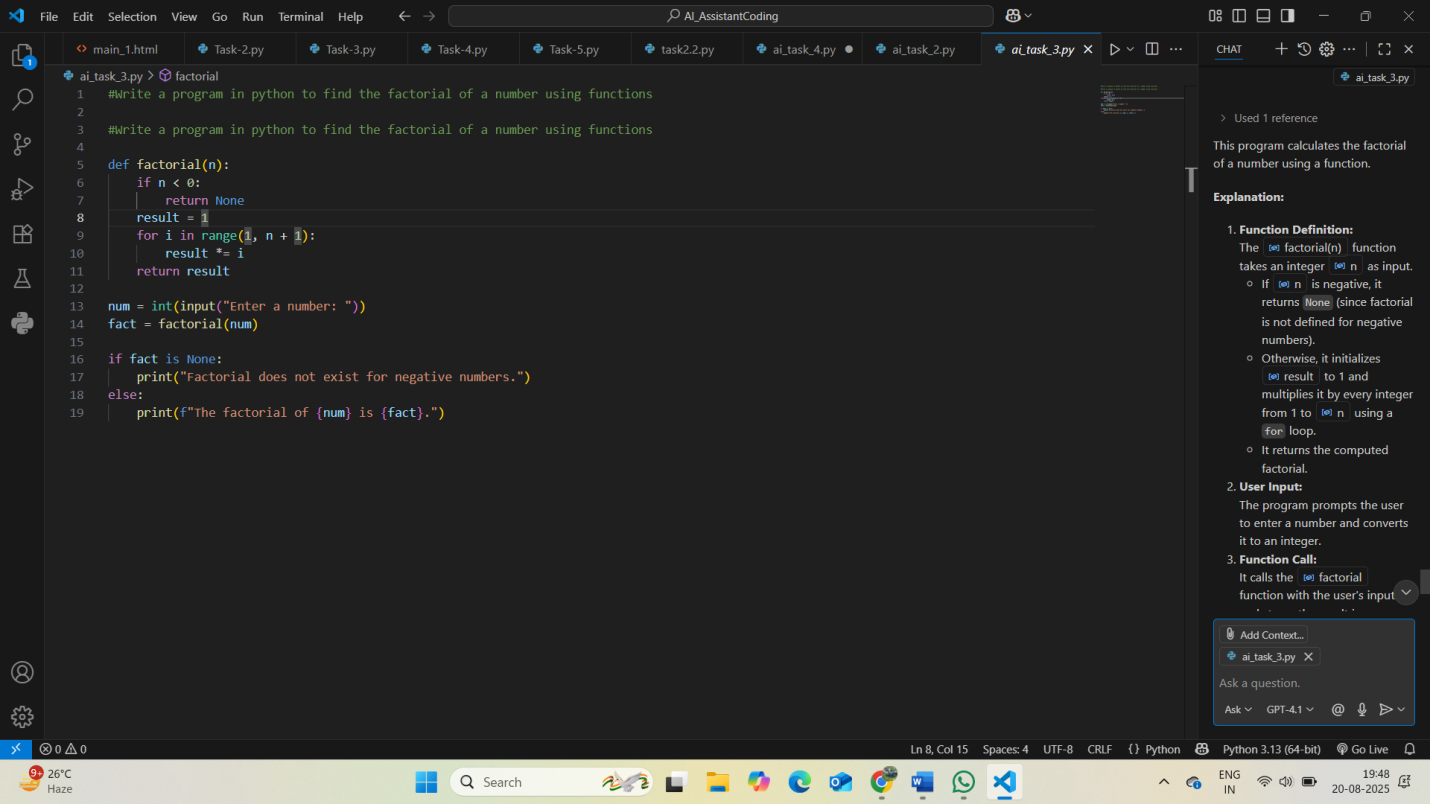
**Explanation:**

You can improve the efficiency and readability of this code in several ways:

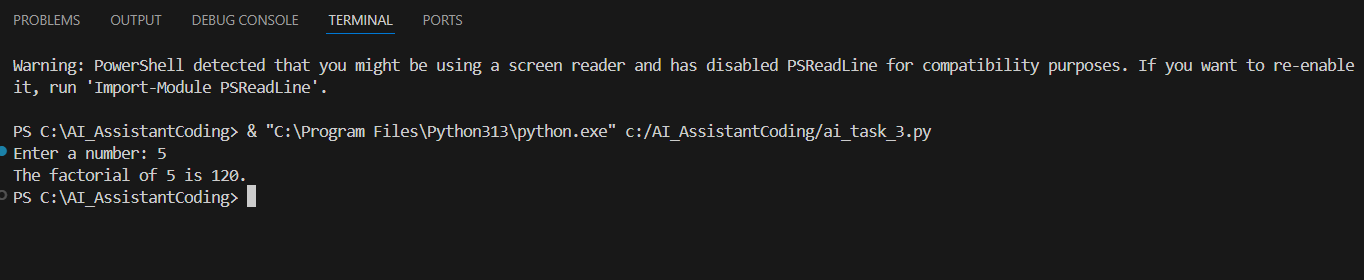
1. Remove unnecessary initialization:  
   You can move the factorial = 1 assignment inside the else block, since it's not needed for negative or zero cases.
2. Use the math.factorial() function:  
   Python’s standard library provides a highly optimized factorial function. This is the most efficient and concise way.
3. Use a one-liner for the loop (if you want to avoid math.factorial):  
   You can use prod() from math (Python 3.8+) or a generator expression with reduce.

**TASK-3:** Generate a Python program that calculates the factorial of a number using a user-defined function.

**Code:**



**Output:**

****

**Explanation:**

This program calculates the factorial of a number using a function.

**Explanation:**

1. **Function Definition:**  
   The [factorial(n)](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) function takes an integer [n](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) as input.
   * If [n](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) is negative, it returns None (since factorial is not defined for negative numbers).
   * Otherwise, it initializes [result](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) to 1 and multiplies it by every integer from 1 to [n](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) using a for loop.
   * It returns the computed factorial.
2. **User Input:**  
   The program prompts the user to enter a number and converts it to an integer.
3. **Function Call:**  
   It calls the [factorial](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) function with the user's input and stores the result in [fact](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
4. **Output:**
   * If [fact](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) is None, it prints a message that factorial does not exist for negative numbers.
   * Otherwise, it prints the factorial of the entered number.

**Example:**  
If the user enters 4, the output will be:  
The factorial of 4 is 24.

If the user enters -2, the output will be:  
Factorial does not exist for negative numbers.

**TASK-4:** Differentiate between the factorial program with functions and without functions in terms of logic, reusability, and execution.

**Comparison Table:**

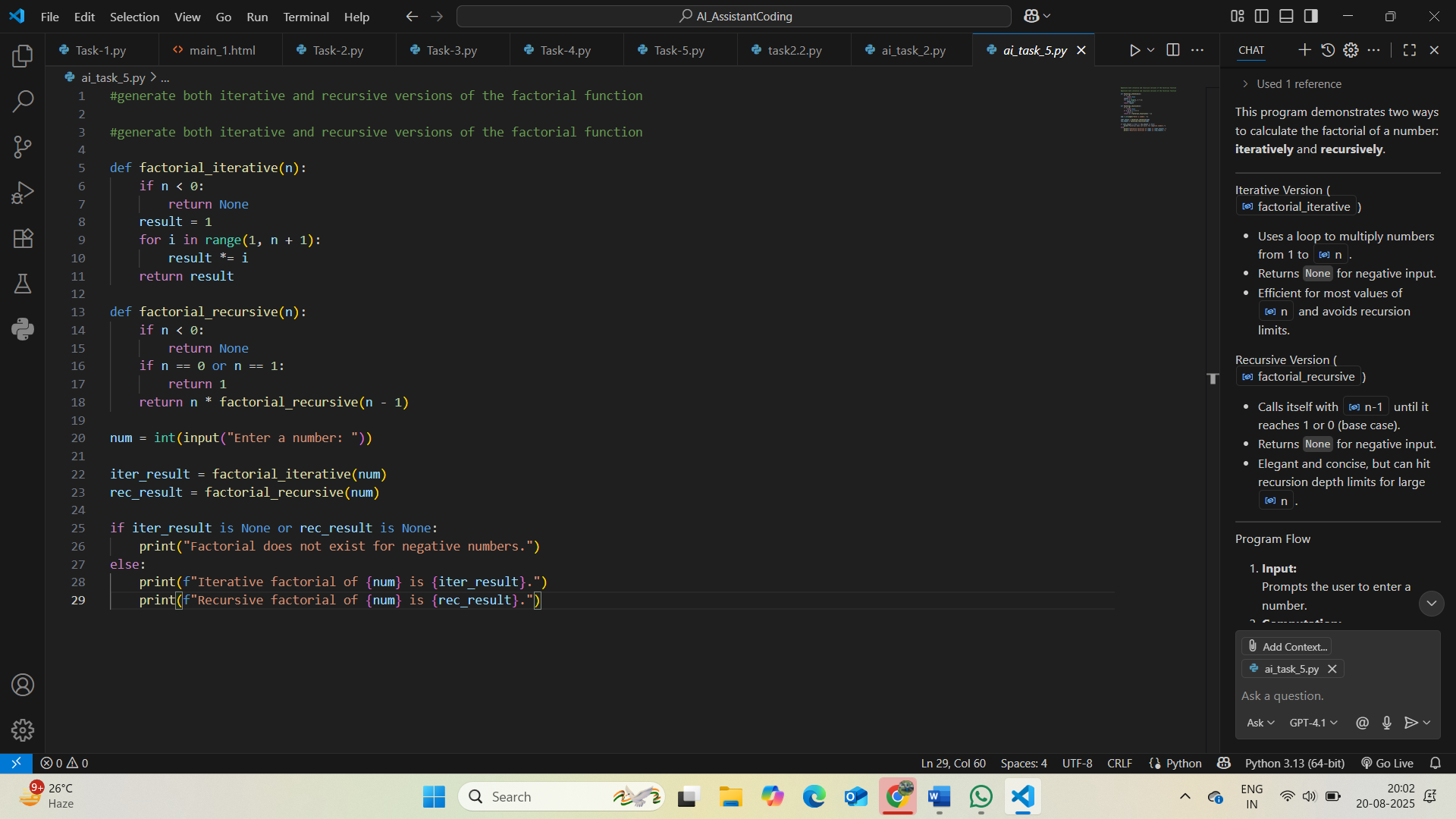
* **Difference between factorial program with functions and without functions:**

| **Aspect** | **With Functions (def factorial(n): ...)** | **Without Functions (direct code)** |
| --- | --- | --- |
| **Logic** | **Logic is encapsulated inside a function. The main code calls this function with the required argument.** | **Logic is written directly in the main program, usually inside a loop or conditional block.** |
| **Reusability** | **Highly reusable. The function can be called multiple times with different inputs, or imported into other programs.** | **Not reusable. The code executes only once for the given input; to use it again, you must copy-paste or rewrite the logic.** |
| **Execution** | **The function is defined once and can be executed any number of times by calling it. Cleaner and more modular.** | **The code executes sequentially from top to bottom. No modularity; harder to maintain or extend.** |

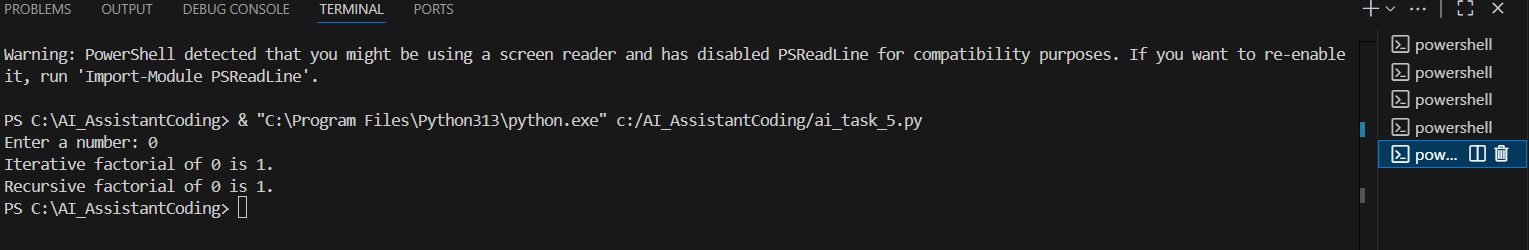
**Summary:**Using functions improves code organization, reusability, and maintainability, while direct code is simpler but less flexible and reusable.

**TASK-5:** Generate both iterative and recursive versions of the factorial function.

**Code:**



**Output:**

****

**Explanation:**

This program demonstrates two ways to calculate the factorial of a number: **iteratively** and **recursively**.

**Iterative Version (**[factorial\_iterative](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)**)**

* Uses a loop to multiply numbers from 1 to [n](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
* Returns None for negative input.
* Efficient for most values of [n](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) and avoids recursion limits.

**Recursive Version (**[factorial\_recursive](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)**)**

* Calls itself with [n-1](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) until it reaches 1 or 0 (base case).
* Returns None for negative input.
* Elegant and concise, but can hit recursion depth limits for large [n](vscode-file://vscode-app/c:/Users/sweety/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).

**Program Flow**

1. **Input:**  
   Prompts the user to enter a number.
2. **Computation:**  
   Calculates factorial using both iterative and recursive functions.
3. **Output:**
   * If the input is negative, prints an error message.
   * Otherwise, prints both the iterative and recursive results.

**Summary:**

* Both methods return the same result for valid input.
* The iterative approach is generally more efficient for large numbers in Python.
* The recursive approach is more mathematical and elegant for small numbers.