# AI ASSISTED CODING

## TASK 2:

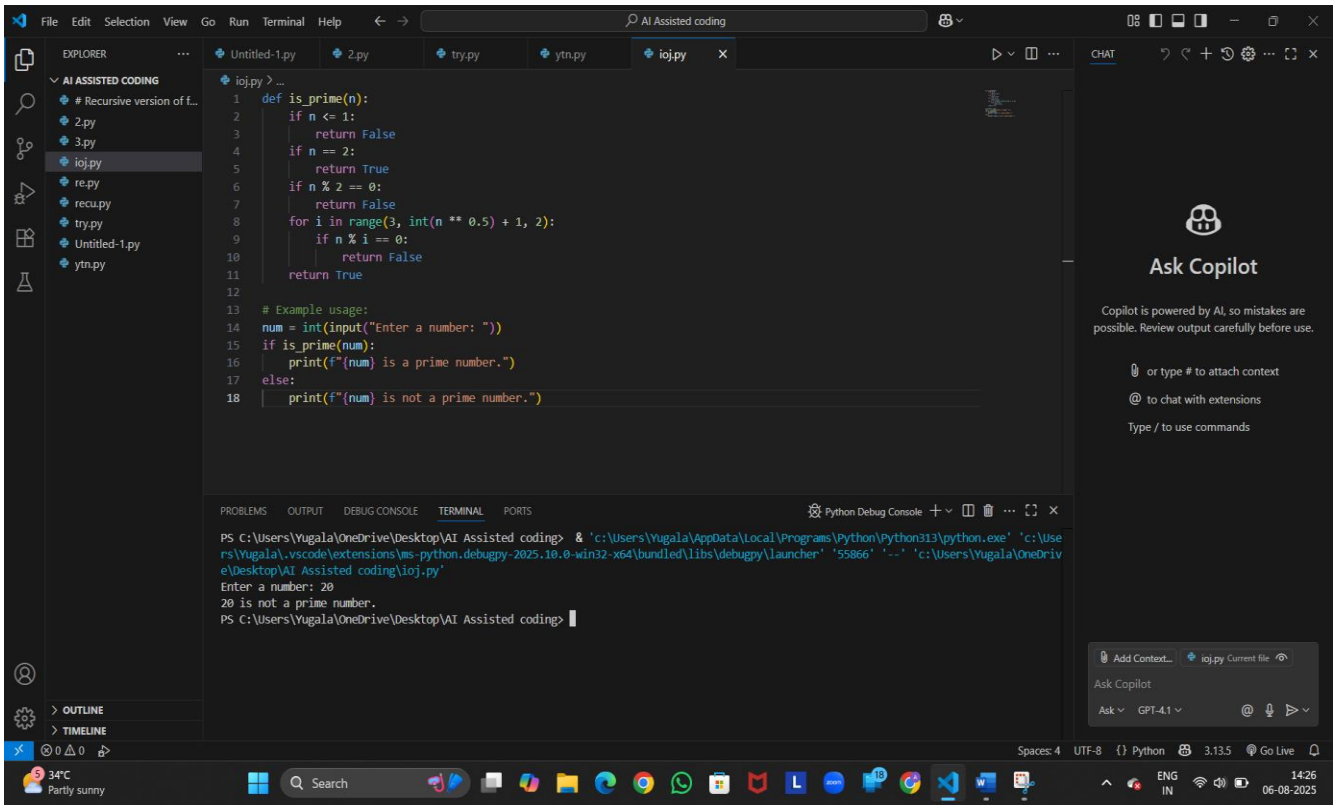
>> Use copilot to generate a is_prime() python function.

**prompt:**

Write a python code to check whether a number is prime or not.

**Expected output:**

Function to check primality with correct logic.




**Observation:**

A prime number is a number greater than 1 that has no positive divisors other than 1 and itself.

- **Functionality**: Returns True for prime numbers and False otherwise.
- **Suitable for checking primality of large numbers due to reduced iterations.**

## TASK 3:

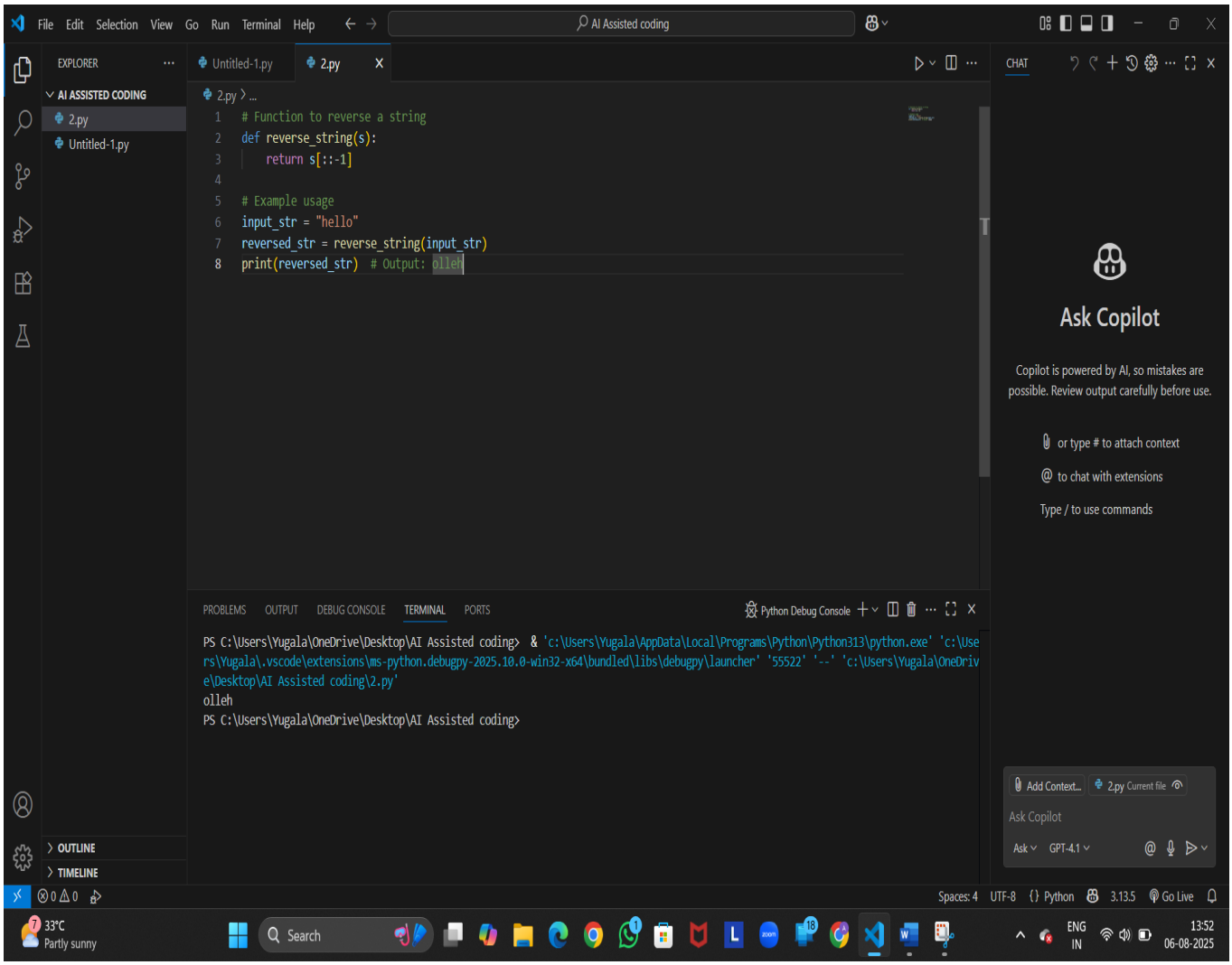
>>Write a comment like # function to reverse a string and use copilot to generate the function.

**Prompt:**

>>Write a python code for comment like # function to reverse a string and use function.

**Expected output:**

>>Auto-completed reverse function.



**Observation:**

- **Logic: Uses Python slicing [::-1] to reverse the string efficiently.**
- **Simplicity: One-liner function; concise and readable.**
- **Functionality: Works for letters, numbers, symbols, and even empty strings.**

## TASK 4:

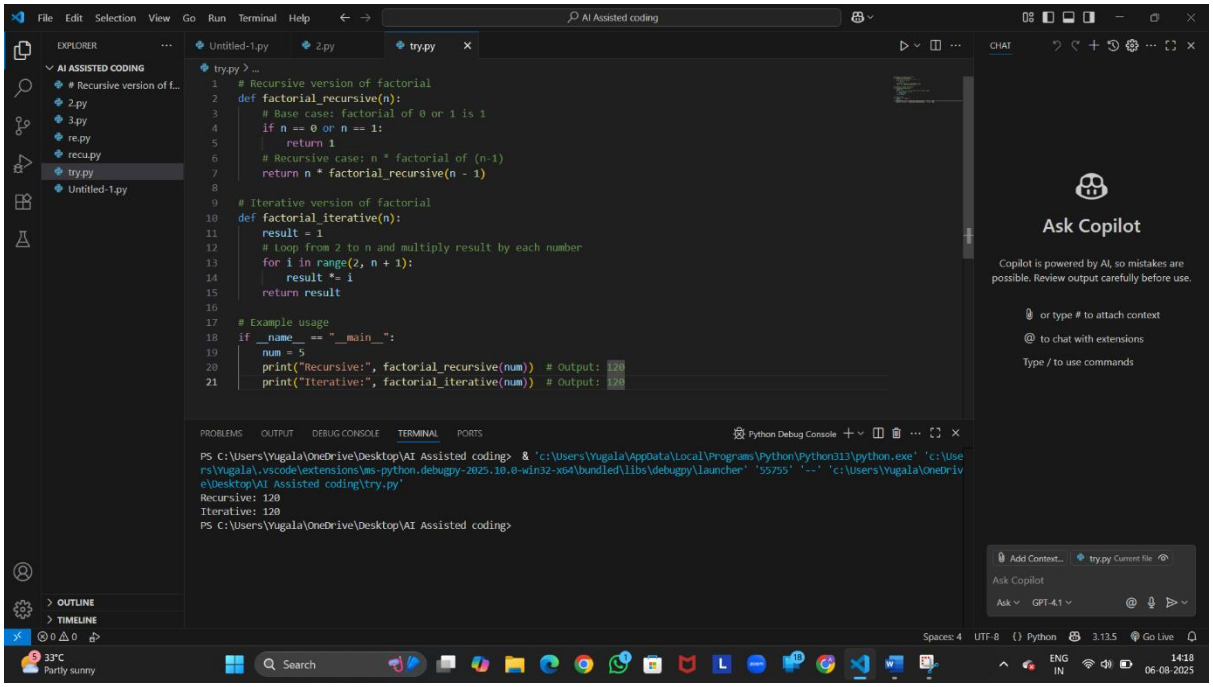
>>Generate both recursive and iterative version of a factorial using comments.

**Prompt:**

>>Write a python to generate both recursive and iterative version of a factorial using comments.

**Expected output:**

>>Two working factorial implementations.



**Observation:**

- **Recursive Version:**
  - **Elegant and mirrors the mathematical definition.**
  - **May cause stack overflow for large n due to deep recursion.**
  - **Time complexity: O(n); Space complexity: O(n) (due to call stack).**
- **Iterative Version:**
  - **More memory-efficient and avoids recursion limits.**
  - **Preferred for large values of n.**
  - **Time complexity: O(n); Space complexity: O(1).**
- **Both Implementations:**
  - **Correctly handle base cases (0! = 1, 1! = 1).**
  - **Produce identical results for valid non-negative integers.**

## TASK 5:

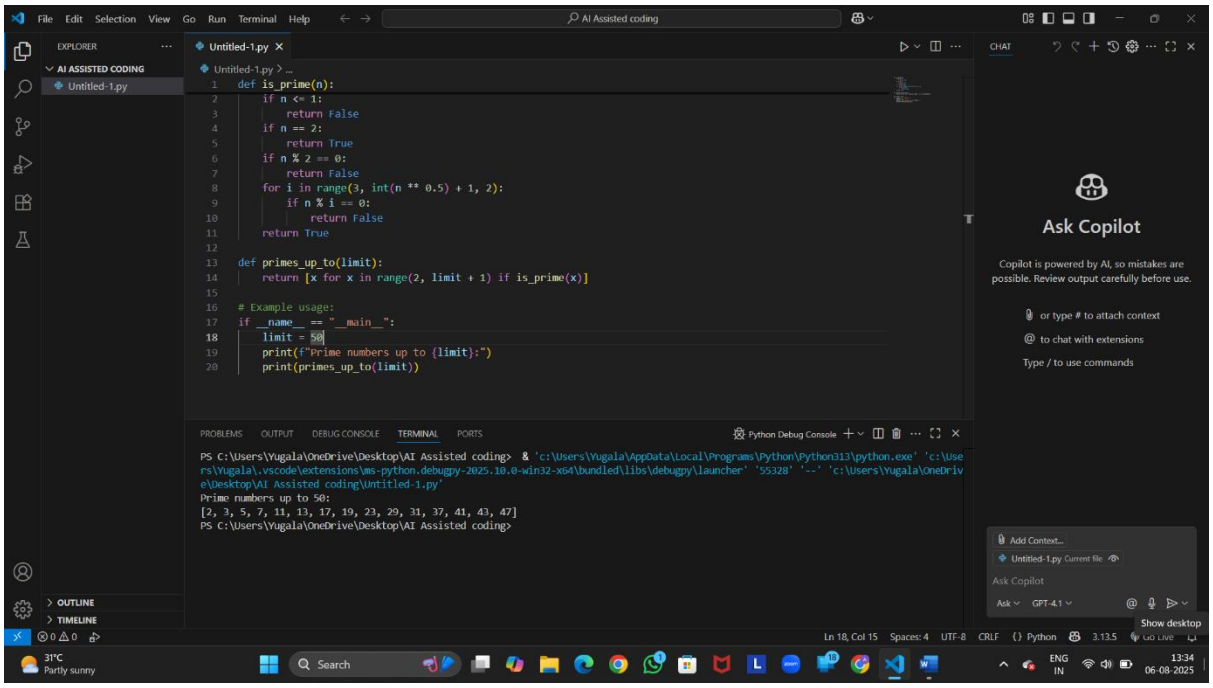
>>Use copilot to find the largest number in a list. Assess code quality and efficiency.

**Prompt:**

>> Write a python code to find the largest number in a list and assess code quality and efficiency.

**Expected output:**

>> A valid function with your review.




## Observation:

- **Correctness: Accurately finds the largest number by comparing each element.**
- **Edge Case Handling: Returns None for an empty list, avoiding errors.**
- **Efficiency:**
  - **Time complexity: O(n) — linear scan through the list.**
  - **Space complexity: O(1) — uses constant extra space.**
- **Code Quality:**
  - **Clear variable naming (largest, numbers).**