# AI ASSISTED CODING

**NAME:Bayana Arjun**

**HALL TICKET NO:2403A510A9**

**ASSIGNMENT : 1.3**

**Lab Objectives:**

- To install and configure GitHub Copilot in Visual Studio Code.
- To explore AI-assisted code generation using GitHub Copilot.
- To analyze the accuracy and effectiveness of Copilot's code suggestions.
- To understand prompt-based programming using comments and code context

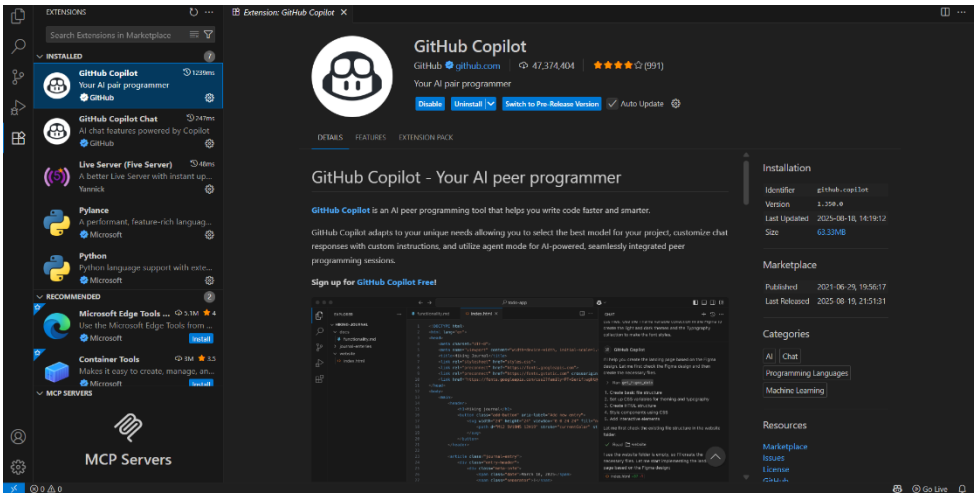
**Task Description#1**

- Install and configure GitHub Copilot in VS Code. Take screenshots of each step.

**Expected Output#1**

- Install and configure GitHub Copilot in VS Code. Take screenshots of each step.

1

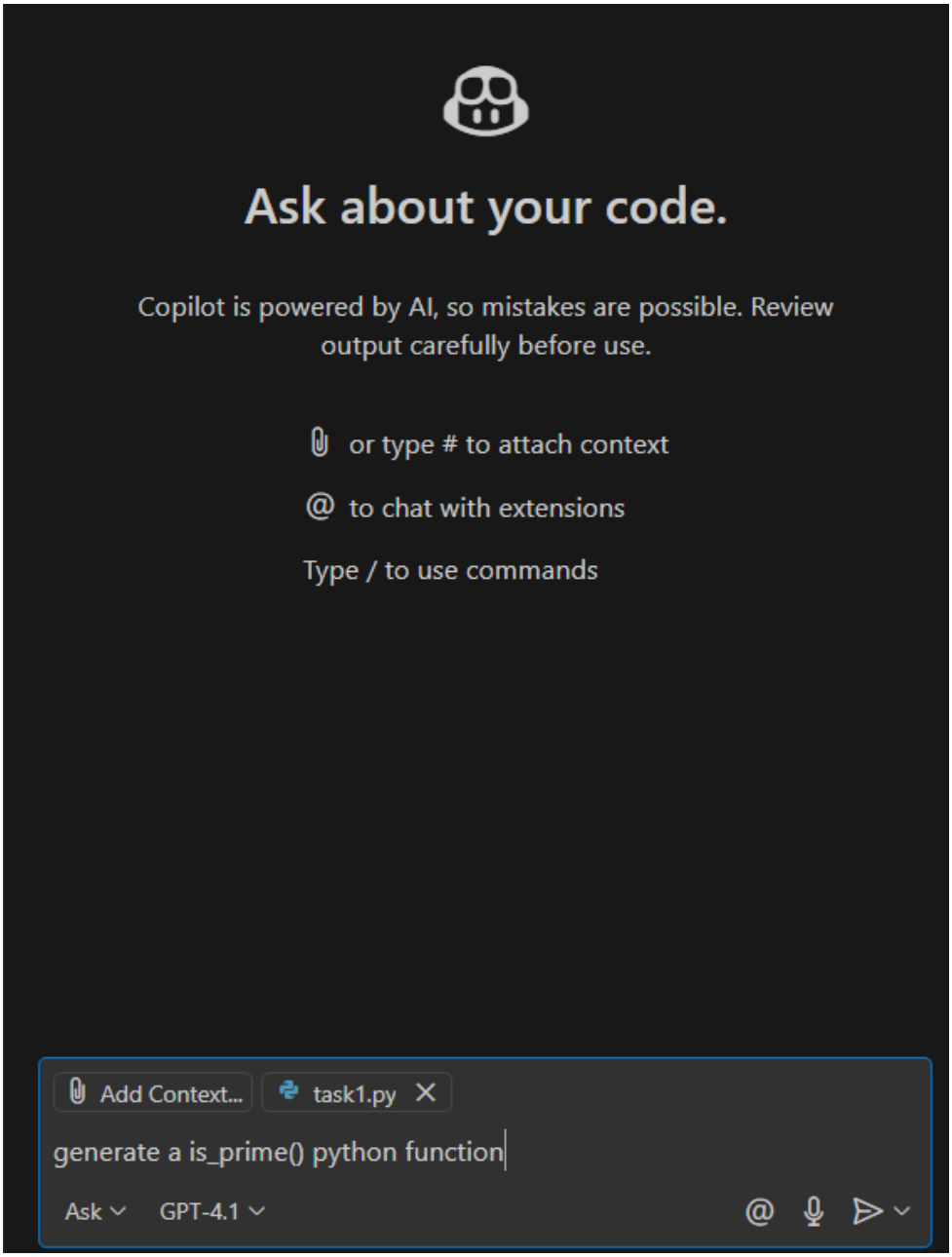



**Task Description#2**

- Use Copilot to generate a is_prime() Python function.

**Expected Output#2**

- Function to check primality with correct logic.



generate a is_prime() python function

task1.py

Used 1 reference

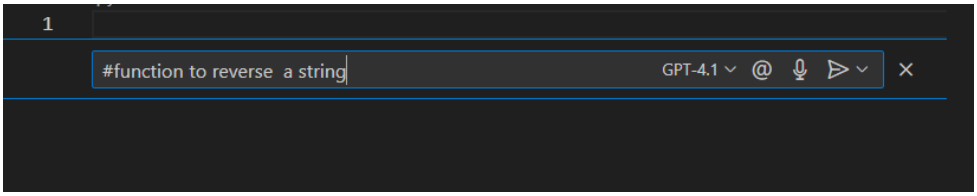
```python
def is_prime(n):
    """Check if a number is prime."""
    if n <= 1:
        return False
    if n == 2:
        return True
    if n % 2 == 0:
        return False
    for i in range(3, int(n ** 0.5) + 1, 2):
        if n % i == 0:
            return False
    return True
```

**Task Description#3**

- Write a comment like # Function to reverse a string and use Copilot to generate the function.

**Expected Output#3**

- Auto-completed reverse function




```python
def reverse_string(s):
    return s[::-1]
```

**Task Description#4**

- Generate both recursive and iterative versions of a factorial function using comments..

**Expected Output#4**

- Two working factorial implementations

```python
# Recursive version of factorial
def factorial_recursive(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial_recursive(n - 1)

# Iterative version of factorial
def factorial_iterative(n):
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result
```

**Task Description#5**

- Use Copilot to find the largest number in a list. Assess code quality and efficiency.

**Expected Output#5**

- A valid function with your review

find the largest number in a list. Assess code quality and efficiency.

task1.py

> Used 1 reference

To find the largest number in a list, you can use Python's built-in `max()` function, which is efficient and concise. Alternatively, you can implement it manually:

```
def find_largest(numbers):
    """Return the largest number in a list."""
    if not numbers:
        raise ValueError("List is empty")
    largest = numbers[0]
    for num in numbers[1:]:
        if num > largest:
            largest = num
    return largest
```

**Assessment of code quality and efficiency:**

- The built-in `max()` is optimal for readability and performance.
- The manual approach above is clear, handles empty lists, and runs in O(n) time.
- Both methods are efficient for typical use cases.
- Code is readable, maintainable, and follows Python conventions.
- For very large lists, both approaches are suitable unless memory is constrained.

**Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots**

**Evaluation Criteria:**

| Criteria | Max Marks |
|---|---|
| Successful Setup of Copilot (Task #1) | 0.5 |
| is_prime() Python function (Task #2) | 0.5 |
| Reverse a string function (Task #3) | 0.5 |
| Factorial Function (Task #4) | 0.5 |
| Find the largest number (Task #5) | 0.5 |
| **Total** | **2.5 Marks** |