

AI-ASSISTED CODING

ASSIGNMENT-1.3

Name:A.Dhanalaxmi

Rollno:2403A51269

Task Description#1

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

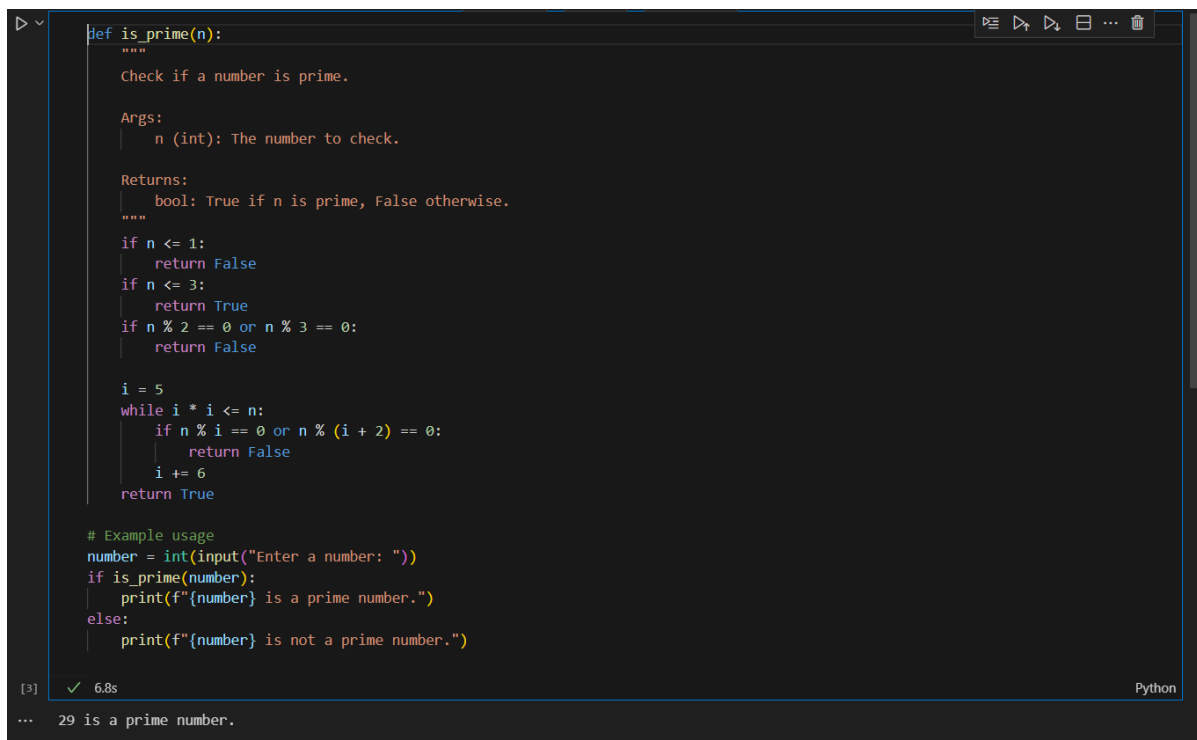
Expected Output#1

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

Task2:

generate a is_prime() Python function,and send the code

Code with output:



```
def is_prime(n):
    """
    Check if a number is prime.

    Args:
        n (int): The number to check.

    Returns:
        bool: True if n is prime, False otherwise.
    """
    if n <= 1:
        return False
    if n <= 3:
        return True
    if n % 2 == 0 or n % 3 == 0:
        return False

    i = 5
    while i * i <= n:
        if n % i == 0 or n % (i + 2) == 0:
            return False
        i += 6
    return True

# Example usage
number = int(input("Enter a number: "))
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

[3] ✓ 6.8s Python

... 29 is a prime number.

Task-3:

Prompt:

Write a comment like # Function to reverse a string and also generate the function

Code with output:

```
# Function to reverse a string
def reverse_string(s):
    """
    Reverse the given string.

    Args:
        s (str): Input string to be reversed.

    Returns:
        str: Reversed version of the input string.
    """
    return s[::-1]

# Example usage with output
input_str = input("Enter a string: ")
reversed_str = reverse_string(input_str)
print(f"Reversed string: {reversed_str}")
```

[5] ✓ 4.1s Python

... Reversed string: unos

Task-4:

Prompt:

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

Code with output:

 Recursive Version:

```
# Recursive function to calculate factorial
def factorial_recursive(n):
    """
    Recursively calculates the factorial of a non-negative integer.

    Args:
        n (int): The number to compute factorial for.

    Returns:
        int: Factorial of n.
    """
    if n < 0:
        raise ValueError("Factorial is not defined for negative numbers.")
    if n == 0 or n == 1:
        return 1
    return n * factorial_recursive(n - 1)

# Output
num = int(input("Enter a number for recursive factorial: "))
print(f"Factorial (recursive) of {num} is {factorial_recursive(num)}")
```

[7] ✓ 4.1s Python

... Factorial (recursive) of 5 is 120

Iterative Factorial Function:

```
# Iterative function to calculate factorial
def factorial_iterative(n):
    """
    Iteratively calculates the factorial of a non-negative integer.

    Args:
        n (int): The number to compute factorial for.

    Returns:
        int: Factorial of n.
    """
    if n < 0:
        raise ValueError("Factorial is not defined for negative numbers.")

    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Output
num = int(input("Enter a number for iterative factorial: "))
print(f"Factorial (iterative) of {num} is {factorial_iterative(num)}")
```

[8] ✓ 10.8s Pyth

... Factorial (iterative) of 5 is 120

Task-5:

Prompt:

send code for finding the largest number in a list and Assess code quality and efficiency. with a valid function with your review

Code with output:

```
# Function to find the largest number in a list
def find_max(numbers):
    """
    Returns the largest number in a list.

    Args:
        numbers (list): A list of numeric values.

    Returns:
        float or int: The largest number in the list.

    Raises:
        ValueError: If the list is empty.
    """
    if not numbers:
        raise ValueError("Input list must not be empty.")

    max_value = numbers[0]
    for num in numbers[1:]:
        if num > max_value:
            max_value = num
    return max_value

# Example usage
data = [12, 45, 3, 67, 23, 89, 34]
print(f"Largest number in the list: {find_max(data)}")
```

[9] ✓ 0.0s Pyth

... Largest number in the list: 89