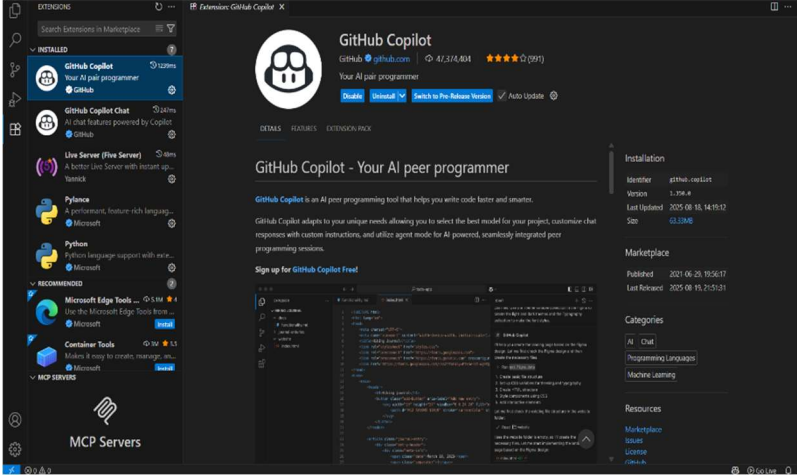


	<div>AI ASSISTED CODING</div> <div>NAME:J.KEERTHI PRIYA</div> <div>ROLL NO:2403A510G4</div> <div>ASSIGNMENT:1.1</div>	
1	<div>Task 0</div> <div><ul style="list-style-type: none">● Install and configure GitHub Copilot in VS Code. Take screenshots of each step.</div> <div>Expected Output</div> <div><ul style="list-style-type: none">● Install and configure GitHub Copilot in VS Code. Take screenshots of each step.</div> <div></div> <div>Task 1: Factorial without Functions</div> <div><ul style="list-style-type: none">● Description: Use GitHub Copilot to generate a Python program that calculates the factorial of a number without defining any functions (using loops directly in the main code).● Expected Output:<ul style="list-style-type: none">○ A working program that correctly calculates the factorial for user-provided input.○ Screenshots of the code generation process.</div>	



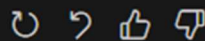
WRITE a Python program that calculates the factorial of a number without defining any functions (using loops directly in the main code).

Untitled-3

> Used 1 reference

```
num = int(input("Enter a number: "))
result = 1

if num < 0:
    print("Factorial is not defined for negative numb
elif num == 0 or num == 1:
    print("Factorial:", 1)
else:
    for i in range(2, num + 1):
        result *= i
    print("Factorial:", result)
```



GPT-4.1 • 1x



Add Context...



Untitled-3 X

Ask a question.

Ask ▾

GPT-4.1 ▾



Spaces: 4 UTF-8 CRLF {} Plain Text Go Live

Task 2: Improving Efficiency

- Description:
Examine the Copilot-generated code from Task 1 and demonstrate

	<p>how its efficiency can be improved (e.g., removing unnecessary variables, optimizing loops).</p> <ul style="list-style-type: none">• Expected Output:<ul style="list-style-type: none">○ Original and improved versions of the code.○ Explanation of how the improvements enhance performance. <p>ORIGINAL CODE:</p>	
--	---	--



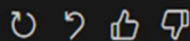
WRITE a Python program that calculates the factorial of a number without defining any functions (using loops directly in the main code).

Untitled-3

> Used 1 reference

```
num = int(input("Enter a number: "))
result = 1

if num < 0:
    print("Factorial is not defined for negative numb
elif num == 0 or num == 1:
    print("Factorial:", 1)
else:
    for i in range(2, num + 1):
        result *= i
    print("Factorial:", result)
```



GPT-4.1 • 1x



Add Context...



Untitled-3 X

Ask a question.

Ask ▾

GPT-4.1 ▾



Spaces: 4

UTF-8

CRLF

{ } Plain Text



Go Live



Go Live



IMPROVED VERSION OF CODE:

```

num = int(input("Enter a number: "))

if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    result = 1
    for i in range(2, num + 1):
        result *= i
    print("Factorial:", result)

```

Explanation of Improvements:

- The elif block checking for num == 0 or num == 1 is removed because the loop naturally handles these cases by not executing any multiplication when num is 0 or 1.
- The result = 1 initialization is only done when needed, avoiding it when the input is negative.
- Overall, the structure is cleaner and still easy to understand.

Task 3: Factorial with Functions

- Description:
Use GitHub Copilot to generate a Python program that calculates the factorial of a number using a user-defined function.
- Expected Output:
 - Correctly working factorial function with sample outputs.
 - Documentation of the steps Copilot followed to generate the function.

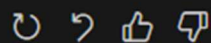


to generate a Python program that calculates the factorial of a number using a user-defined function.

num = int(input("Enter a number: "))

> Used 1 reference

```
def factorial(n):  
    """Calculate factorial of a number."""  
    if n < 0:  
        return None  
    result = 1  
    for i in range(2, n + 1):  
        result *= i  
    return result  
  
num = int(input("Enter a number: "))  
fact = factorial(num)  
if fact is None:  
    print("Factorial is not defined for negative numb  
else:  
    print("Factorial:", fact)
```



Add Context...

num = int(input("Enter a number: "))

Ask a question.

Ask ▾ GPT-4.1 ▾



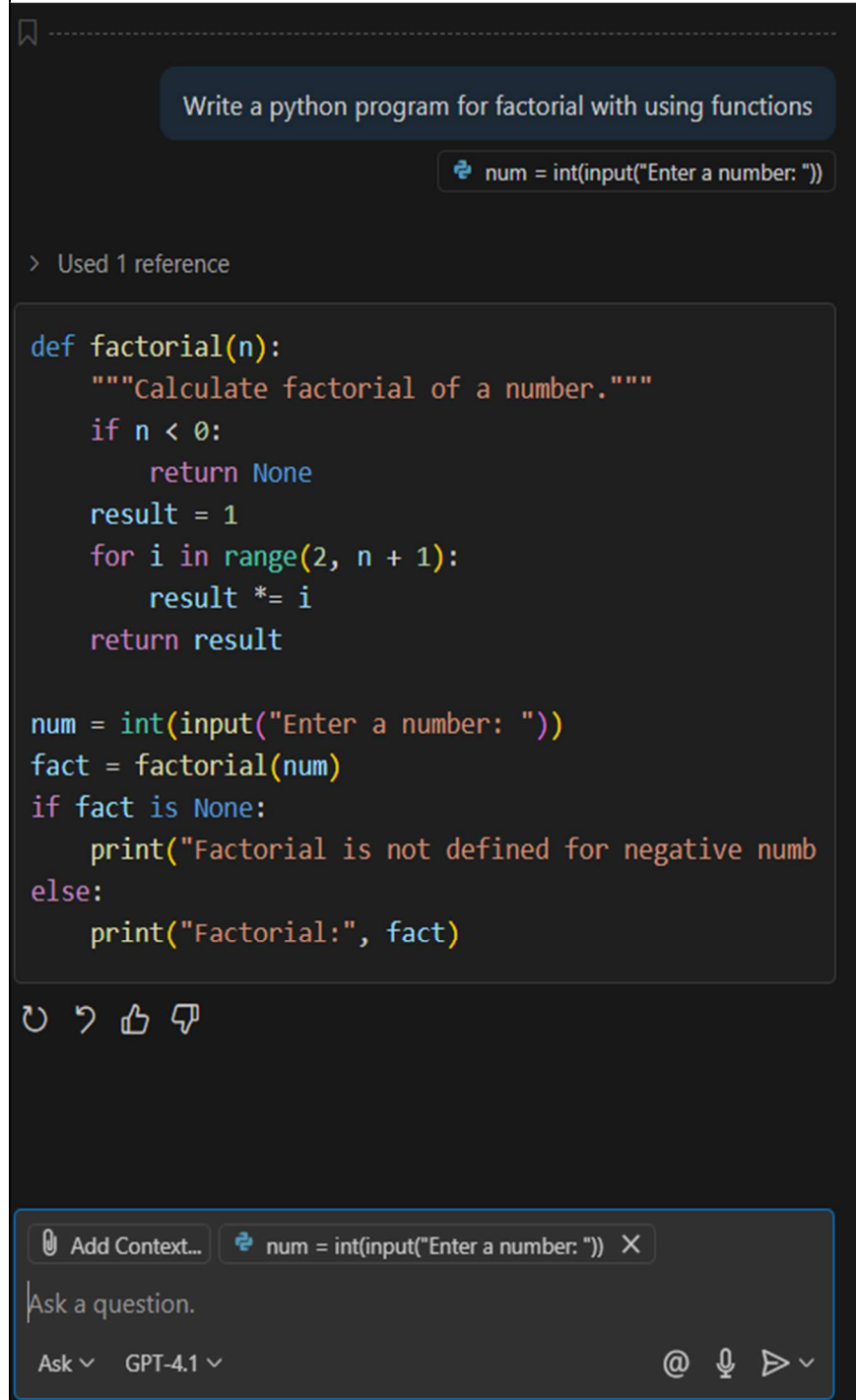
Task 4: Comparative Analysis – With vs Without Functions

- Description:
Differentiate between the Copilot-generated factorial program with functions and without functions in terms of logic, reusability, and

execution.

- Expected Output:
 - A comparison table or short report explaining the differences.

Factorial program using functions :



The screenshot shows a Python IDE with a dark theme. At the top, there is a prompt: "Write a python program for factorial with using functions". Below this, a code editor contains the following Python code:

```
num = int(input("Enter a number: "))

> Used 1 reference

def factorial(n):
    """Calculate factorial of a number."""
    if n < 0:
        return None
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

num = int(input("Enter a number: "))
fact = factorial(num)
if fact is None:
    print("Factorial is not defined for negative numb")
else:
    print("Factorial:", fact)
```

Below the code editor, there are icons for undo, redo, like, and share. At the bottom, there is a search bar with the text "Add Context..." and a dropdown menu showing "num = int(input('Enter a number: '))". Below the search bar, there is a text input field with the placeholder text "Ask a question." and a dropdown menu showing "Ask" and "GPT-4.1".

Factorial program without using functions:

```

task3 > ...
1  # Input: Get number from user
2  number = int(input("Enter a number to calculate factorial: "))
3
4  # Reusability logic: factorial calculation block
5  start = 2          # Starting point of multiplication
6  end = number + 1    # End point for range
7  factorial = 1       # Variable to store result
8
9  # Check for valid input
10 if number < 0:
11     print("Factorial is not defined for negative numbers.")
12 else:
13     for i in range(start, end):
14         factorial *= i # Multiply and accumulate
15     print("Factorial:", factorial)
16

```

Task 5: Iterative vs Recursive Factorial

- Description:
Prompt GitHub Copilot to generate both iterative and recursive versions of the factorial function.
- Expected Output:
 - Two correct implementations.
 - A documented comparison of logic, performance, and execution flow between iterative and recursive approaches

ITERATIVE VERSION:

```

task3 > ...
1  def factorial_iterative(n):
2      if n < 0:
3          raise ValueError("Factorial is not defined for negative numbers.")
4      result = 1
5      for i in range(2, n + 1):
6          result *= i
7      return result
8

```

RECURSIVE VERSION:

```

task3 > ...
1  def factorial_recursive(n):
2      if n < 0:
3          raise ValueError("Factorial is not defined for negative numbers.")
4      if n == 0 or n == 1:
5          return 1
6      return n * factorial_recursive(n - 1)
7

```

Submission Requirements

1. Generate code for each task with comments.
2. Screenshots of Copilot suggestions.
3. Comparative analysis reports (Task 4 and Task 5).

4. Sample inputs/outputs demonstrating correctness.

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	0.5
Comparative Analysis – With vs Without Functions	1
Iterative vs Recursive Factorial	1
Total	2.5 Marks