SR UNIVERSITY

School of Computer Science and Artificial Intelligence

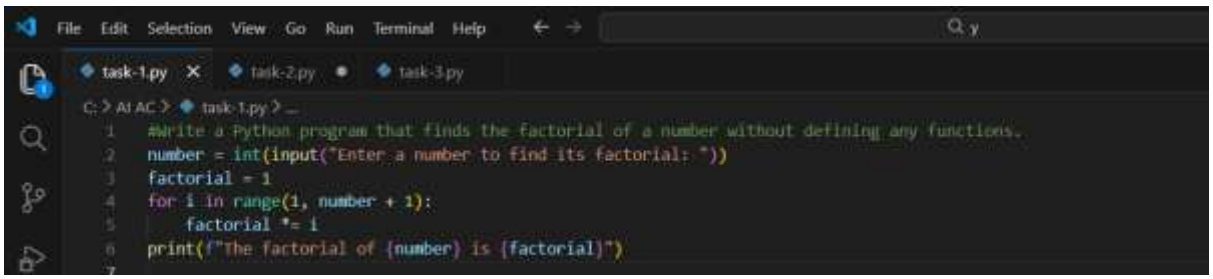
Lab Assignment # 1

Name of Student : GINNE SAI TEJA
Enrollment No. : 2303A51526
Batch No. : 22

Task 1: AI-Generated Logic Without Modularization (Factorial without Functions)

Prompt:-
Write a Python program that finds the factorial of a number without defining any functions.

Code:-



Output:-

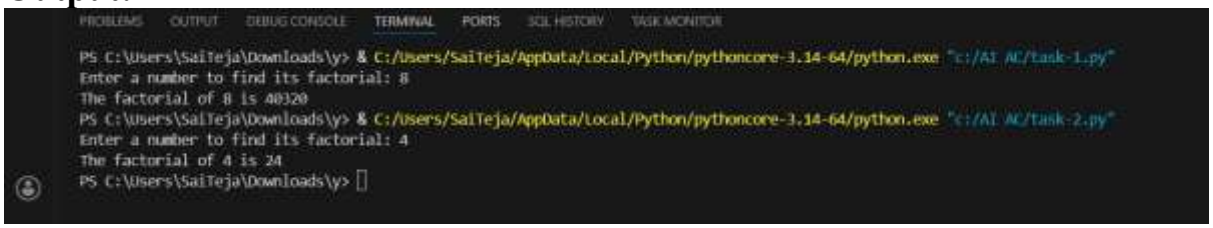
Justification:-
This code employs straightforward step-by-step processing to calculate the factorial without requiring any other constructs. It solves the problem right away and is easy to understand. Nevertheless, because all statements are in a single sequence, this is not the most optimal method for scenarios in which the processing logic has to be replicated or maintained multiple places.

Task 2: AI Code Optimization & Cleanup (Improving Efficiency)

Prompt:-
Refactor the code to eliminate extra variables, streamline the loop, and enhance overall clarity.

Code:-

Output:-



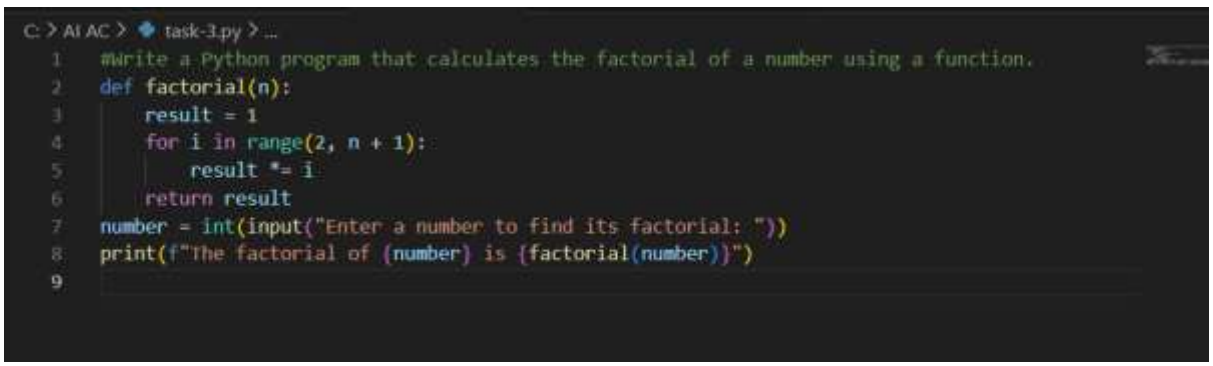
Justification:-

This optimized code optimizes efficiency, eliminating unnecessary steps in the process and compacting the loop logic. By simplifying the process, this reduces the moving parts in the loop, and the resulting code is easy to view and less likely to contain errors, offering a polished, readable piece of code that can easily be applied in a communal project that grows in size..

Task 3: Modular Design Using AI Assistance (Factorial with Functions)

Prompt:-
Write a Python program that calculates the factorial of a number using a function.

Code:-



Output:-

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SQL HISTORY  TASK MONITOR

PS C:\Users\SaiTeja\Downloads\y> & C:/Users/SaiTeja/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/AI AC/task-1.py"
Enter a number to find its factorial: 8
The factorial of 8 is 40320
PS C:\Users\SaiTeja\Downloads\y> & C:/Users/SaiTeja/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/AI AC/task-2.py"
Enter a number to find its factorial: 4
The factorial of 4 is 24
PS C:\Users\SaiTeja\Downloads\y> & C:/Users/SaiTeja/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/AI AC/task-3.py"
Enter a number to find its factorial: 5
The factorial of 5 is 120
PS C:\Users\SaiTeja\Downloads\y>
```

Justification:-

This moves the factorial logic into a function, thus making it modular and easier to manage. A function can then be reused whenever a factorial is needed, rather than having multiple copies of the same code, which can help reduce repetition and errors. This organization also promotes clarity; a program with fewer lines is cleaner and better suited to larger applications or team-based development..

Task 4: Comparative Analysis – Procedural vs Modular AI Code (With vs Without Functions)

Prompt:-
Write a program that compares an AI-written script that's all in one block and a version that uses clear, well-named functions, explaining which is better for understanding, reuse, debugging, big projects, and avoiding over-reliance on AI, and show it in a small table.

Code:-

task4.py > ...

```python
#Write a program that compares an AI-written script that's all in one blo
def calculate_factorial(number):
    if number == 0:
        return 1
    else:
        return number * calculate_factorial(number - 1)

number = int(input("Enter a number to calculate its factorial: "))
result = calculate_factorial(number)
print(f"The factorial of {number} is {result}")
```

Output:-

```
Enter a number to calculate its factorial: 6
The factorial of 6 is 720
PS C:\AIAC LAB>
```

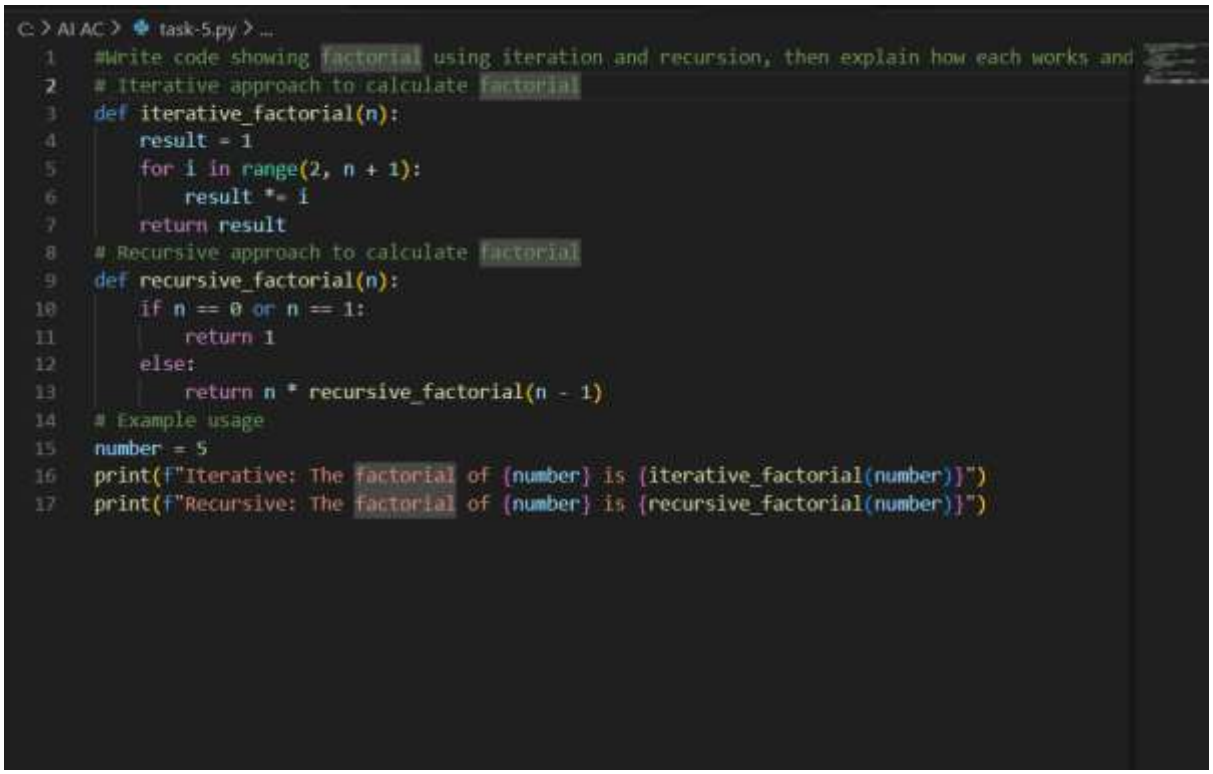
Justification:-

With this breakdown of the task into a neat and tidy recursive function, the code is now much better organized and easier to understand. Rather than being packed off into one long script, this allows the calculation to be kept in a reusable form that is easy to work with.

Task 5: AI-Generated Iterative vs Recursive Thinking

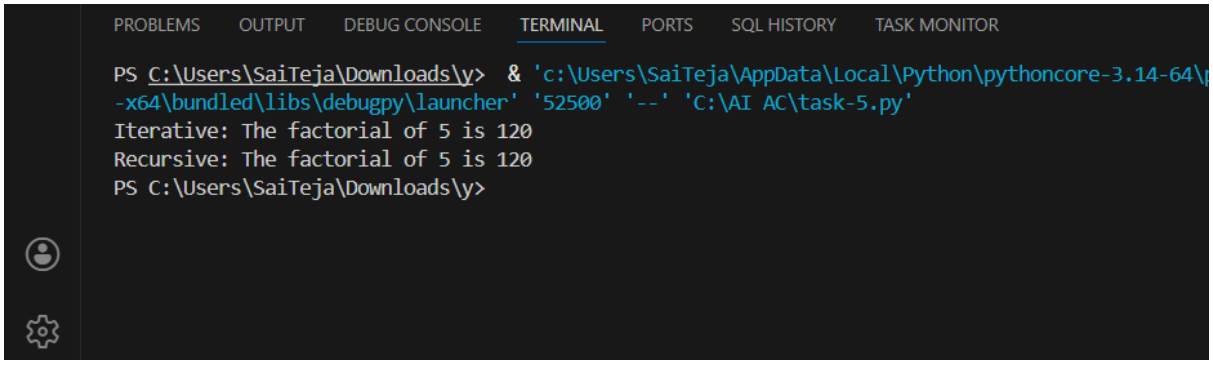
Prompt:-
write a program on create iterative version logic vs recursive version logic of factorial and explain their flow and compare readability, stack usage, performance, and when recursion should be avoided**.**

Code:-



Output:-

SR UNIVERSITY



Justification:-

These two solutions differ in their problem-solving approaches:
A loop is implemented in a step-by-step manner to reach the result incrementally, whereas recursion untangles the problem until reaching the base case. Recursion might turn out easier to understand at first glance, but the loop is always more robust for large inputs since recursion requires fewer calls with significant memory constraints.