AI Assisted Coding — Factorial Assignment

Question 1 — Factorial without using user defined function

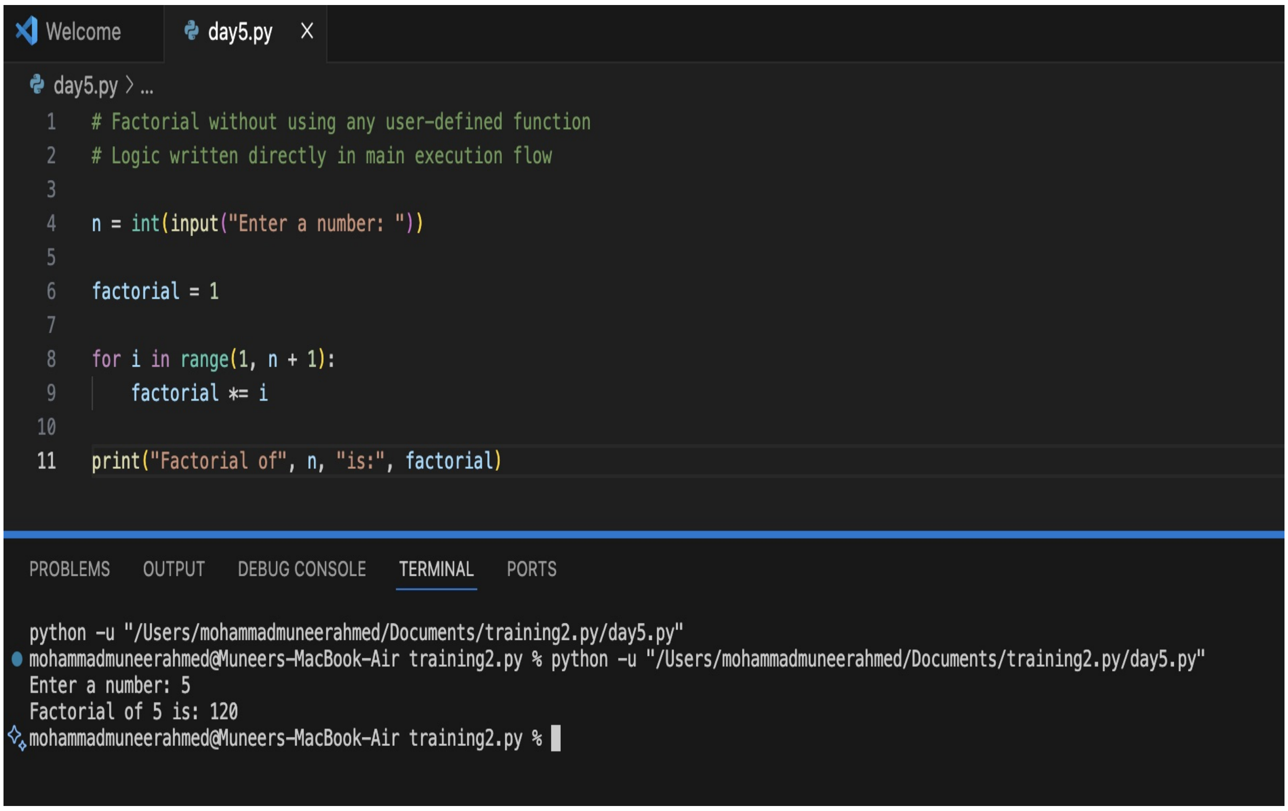
Algorithm

1 Input number
2 Initialize factorial
3 Loop from 1 to n
4 Multiply each step
5 Print result

Pseudo Code

```
START
READ n
fact = 1
FOR i = 1 TO n
 fact = fact * i
PRINT fact
END
```

Execution Screenshot



```python
# Factorial without using any user-defined function
# Logic written directly in main execution flow

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

factorial = 1

for i in range(1, n + 1):
    factorial *= i

print("Factorial of", n, "is:", factorial)
```

```
python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
mohammadmuneerahmed@Muneers-MacBook-Air training2.py % python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
Enter a number: 5
Factorial of 5 is: 120
mohammadmuneerahmed@Muneers-MacBook-Air training2.py %
```

Question 2 — Optimized factorial without functions

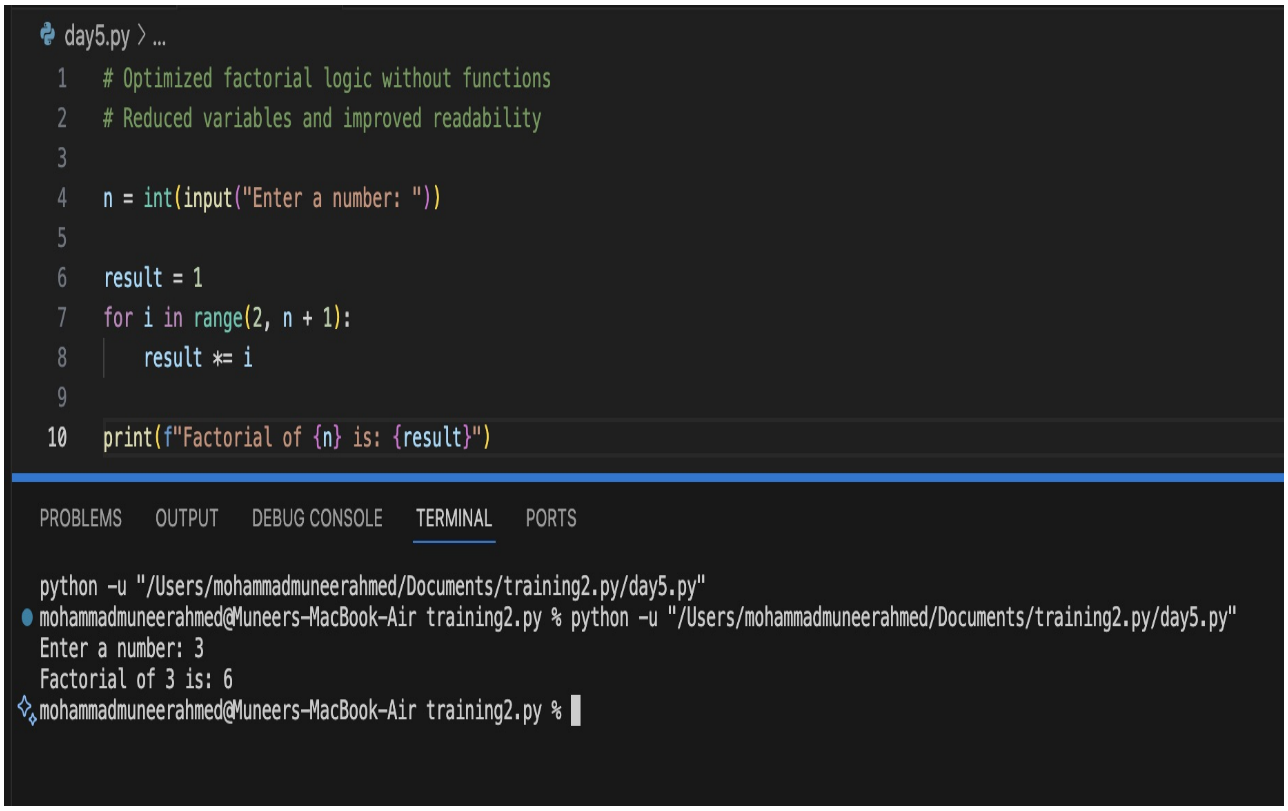
Algorithm

1 Input n
2 Initialize result
3 Loop from 2 to n
4 Multiply
5 Print result

Pseudo Code

```
START
READ n
result = 1
FOR i = 2 TO n
 result = result * i
PRINT result
END
```

Execution Screenshot



```python
# Optimized factorial logic without functions
# Reduced variables and improved readability

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

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

print(f"Factorial of {n} is: {result}")
```

```
python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
mohammadmuneerahmed@Muneers-MacBook-Air training2.py % python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
Enter a number: 3
Factorial of 3 is: 6
mohammadmuneerahmed@Muneers-MacBook-Air training2.py %
```

Question 3 — Factorial using function

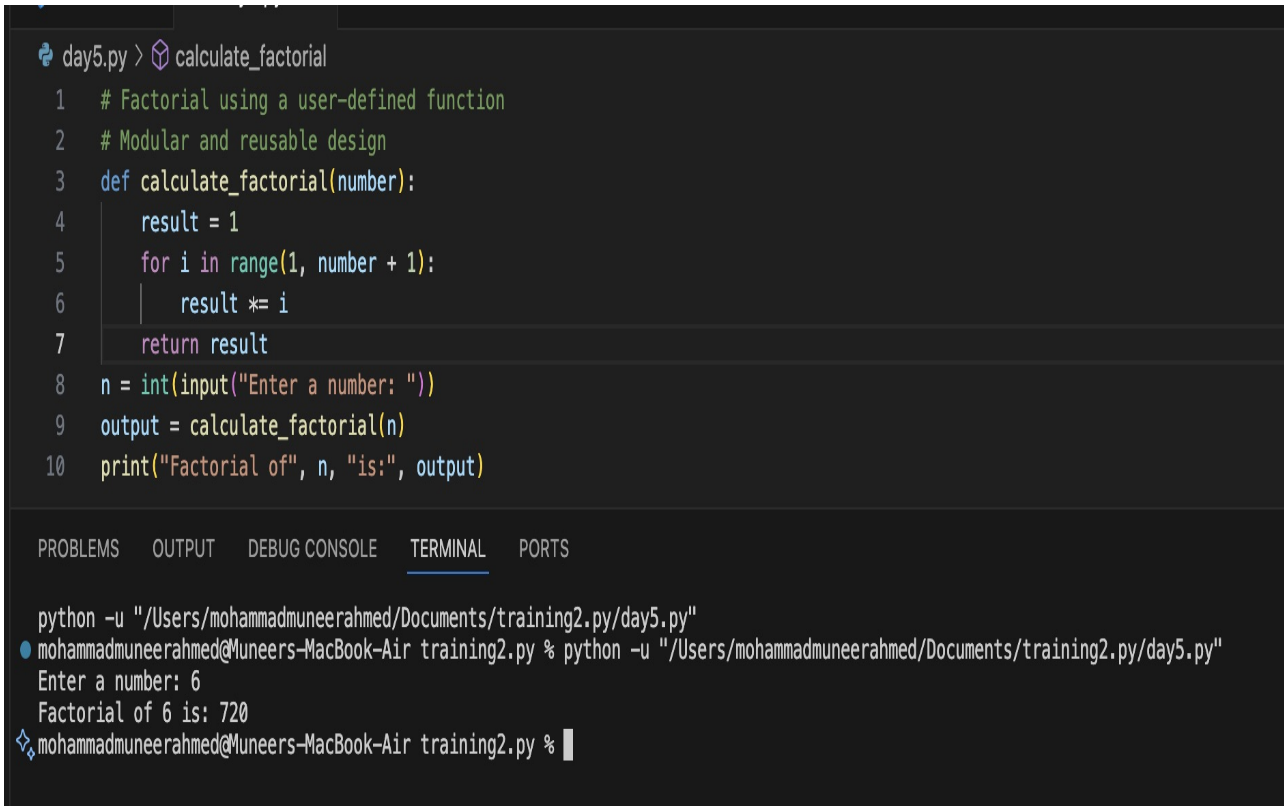
Algorithm

1. Define factorial function
2. Initialize result
3. Loop multiply
4. Return value
5. Call and print

Pseudo Code

```
FUNCTION factorial(n)
 result = 1
 FOR i = 1 TO n
  result = result * i
 RETURN result
```

Execution Screenshot



```python
# Factorial using a user-defined function
# Modular and reusable design
def calculate_factorial(number):
    result = 1
    for i in range(1, number + 1):
        result *= i
    return result
n = int(input("Enter a number: "))
output = calculate_factorial(n)
print("Factorial of", n, "is:", output)
```

```
python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
mohammadmuneerahmed@Muneers-MacBook-Air training2.py % python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
Enter a number: 6
Factorial of 6 is: 720
mohammadmuneerahmed@Muneers-MacBook-Air training2.py %
```

Question 4 — Iterative factorial function

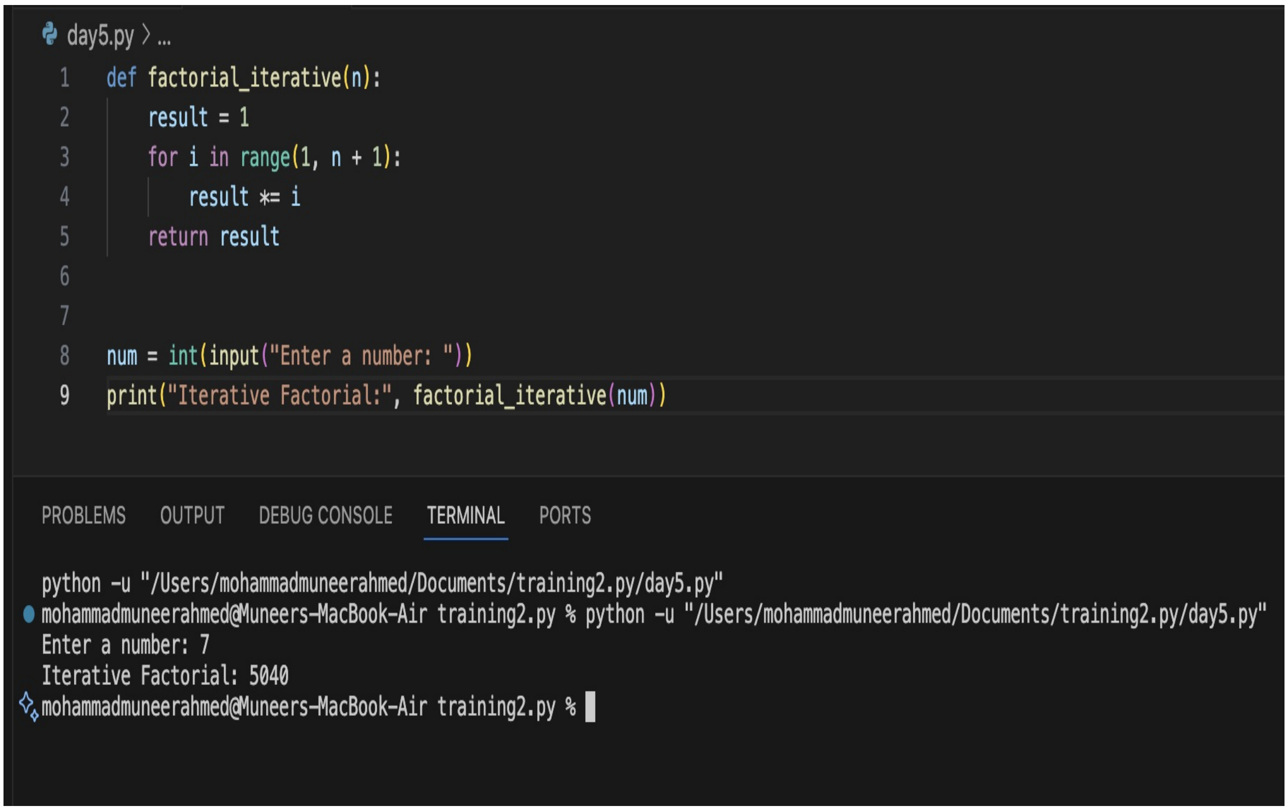
Algorithm

1 Define iterative function
2 Loop multiply
3 Return result

Pseudo Code

```
FUNCTION iterative(n)
 result = 1
 FOR i = 1 TO n
  result = result * i
 RETURN result
```

Execution Screenshot



```python
def factorial_iterative(n):
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result


num = int(input("Enter a number: "))
print("Iterative Factorial:", factorial_iterative(num))
```

```
python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
mohammadmuneerahmed@Muneers-MacBook-Air training2.py % python -u "/Users/mohammadmuneerahmed/Documents/training2.py/day5.py"
Enter a number: 7
Iterative Factorial: 5040
mohammadmuneerahmed@Muneers-MacBook-Air training2.py %
```

Question 5 — Procedural vs Modular comparison

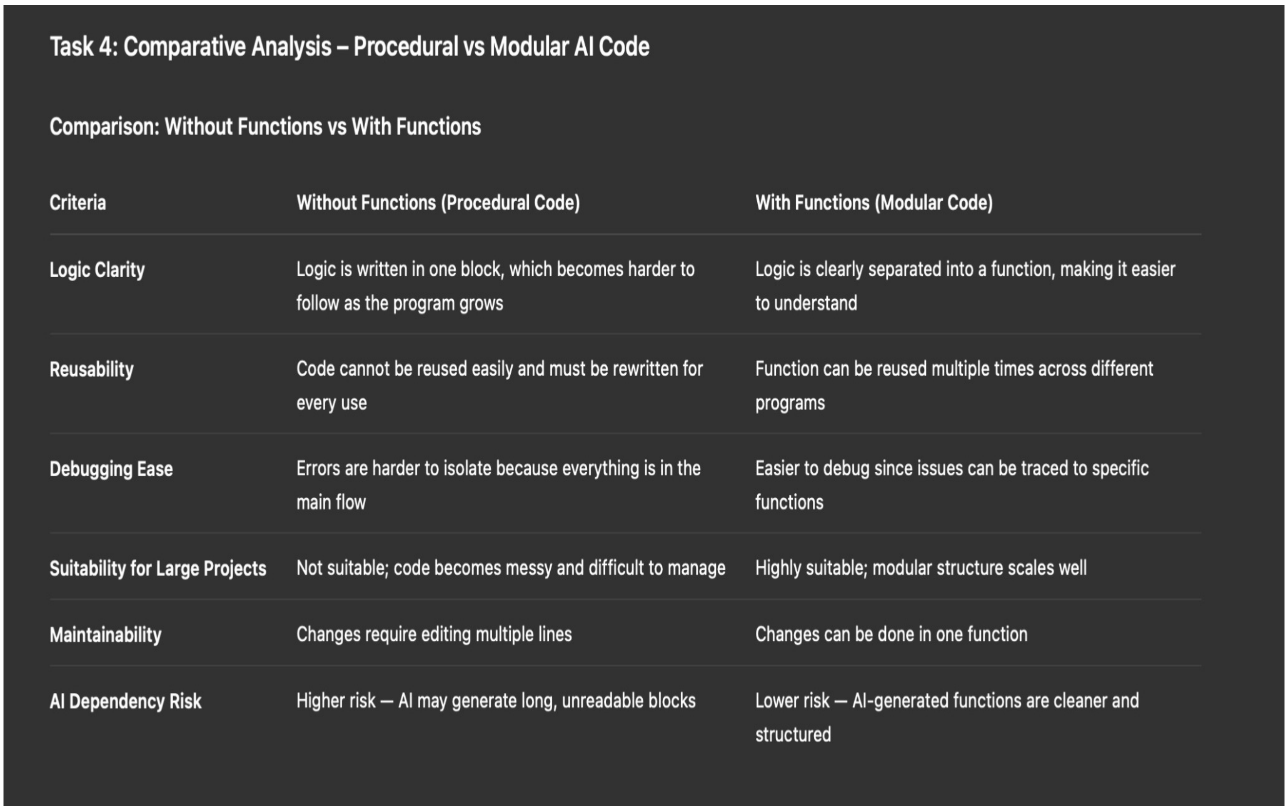
Algorithm

1 Compare clarity
2 Compare reusability
3 Compare debugging
4 Compare scalability

Pseudo Code

```
START
COMPARE procedural vs modular
WRITE differences
END
```

Execution Screenshot



Task 4: Comparative Analysis – Procedural vs Modular AI Code

Comparison: Without Functions vs With Functions

Criteria	Without Functions (Procedural Code)	With Functions (Modular Code)
Logic Clarity	Logic is written in one block, which becomes harder to follow as the program grows	Logic is clearly separated into a function, making it easier to understand
Reusability	Code cannot be reused easily and must be rewritten for every use	Function can be reused multiple times across different programs
Debugging Ease	Errors are harder to isolate because everything is in the main flow	Easier to debug since issues can be traced to specific functions
Suitability for Large Projects	Not suitable; code becomes messy and difficult to manage	Highly suitable; modular structure scales well
Maintainability	Changes require editing multiple lines	Changes can be done in one function
AI Dependency Risk	Higher risk — AI may generate long, unreadable blocks	Lower risk — AI-generated functions are cleaner and structured