

AI ASSISTED CODING

ASSIGNMENT-1.1

M.KEERTHANA

2403A51259

CSE-GEN

BATCH-11

Task 1: Factorial without Functions

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:

- o A working program that correctly calculates the factorial for user-provided input.

- o Screenshots of the code generation process

Prompt: Write a Python program that calculates the factorial of a user-provided non-negative integer using a loop directly in the main code, without defining any functions. Include input validation for negative numbers and print the result.

Code:

```
S-10.4 > ass1.1 > task-1 > ...  
# Factorial calculation without functions  
# Ask user for input  
num = int(input("Enter a non-negative integer: "))  
# Initialize factorial result  
factorial = 1  
  
# Check for valid input  
if num < 0:  
    print("Factorial does not exist for negative numbers.")  
elif num == 0:  
    print("The factorial of 0 is 1.")  
else:  
    # Loop to calculate factorial  
    for i in range(1, num + 1):  
        factorial *= i  
    print(f"The factorial of {num} is {factorial}")
```

Output:

```
PS C:\Users\sonti\OneDrive\Documents\aiass2-9.2> C:\Users\sonti\OneDrive\Documents\aiass2-9.2\AI ASS-10.4\ass1.1\task-1  
Enter a non-negative integer: 0  
The factorial of 0 is 1.  
PS C:\Users\sonti\OneDrive\Documents\aiass2-9.2>
```

Task 2: Improving Efficiency

Description:

Examine the Copilot-generated code from Task 1 and demonstrate how its efficiency can be improved (e.g., removing unnecessary variables, optimizing loops).

● Expected Output:

- o Original and improved versions of the code.

o Explanation of how the improvements enhance performance.
Without Functions.

Prompt: Improve the following Python code that calculates factorial without using functions. Optimize it by removing unnecessary variables and redundant operations, and make the loop more efficient while keeping the output correct.

Code:

Original Copilot-Generated Code

```
-10.4 > ass1.1 > task-2 > ...
# Factorial calculation without functions
num = int(input("Enter a non-negative integer: "))
factorial = 1

if num < 0:
    print("Factorial does not exist for negative numbers.")
elif num == 0:
    print("The factorial of 0 is 1.")
else:
    for i in range(1, num + 1):
        factorial *= i
    print(f"The factorial of {num} is {factorial}")
```

Improved Version

```
-10.4 > ass1.1 > task-2 > ...
# Efficient factorial calculation without functions

num = int(input("Enter a non-negative integer: "))

if num < 0:
    print("Factorial does not exist for negative numbers.")
else:
    factorial = 1
    for i in range(2, num + 1): # Start from 2, multiplying by 1 is redundant
        factorial *= i
    print(f"The factorial of {num} is {factorial}")
```

Output:

```
PS C:\Users\sonti\OneDrive\Documents\ars\sonti\OneDrive\Documents\aiass2-9.2
Enter a non-negative integer: 3
The factorial of 3 is 6
PS C:\Users\sonti\OneDrive\Documents\ars\sonti\OneDrive\Documents\aiass2-9.2
```

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:

- o Correctly working factorial function with sample outputs.
- o Documentation of the steps Copilot followed to generate the function.

Prompt: Write a Python program that calculates the factorial of a user-provided non-negative integer using a user-defined function. Include input validation for negative numbers and print the result. Provide clear comments.

Code:

```
# Factorial calculation using a user-defined function
Tabnine | Edit | Test | Explain | Document
def factorial(n):
    """
    Calculate the factorial of a non-negative integer n.

    Parameters:
    n (int): Non-negative integer

    Returns:
    int: Factorial of n
    """
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Main program
num = int(input("Enter a non-negative integer: "))

if num < 0:
    print("Factorial does not exist for negative numbers.")
else:
    print(f"The factorial of {num} is {factorial(num)}")
```

Output:

```
PS C:\Users\sonti\OneDrive\Documents\aiassers\sonti\OneDrive\Documents\aiass2-9.2/
Enter a non-negative integer: 5
The factorial of 5 is 120
PS C:\Users\sonti\OneDrive\Documents\aiass
```

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:

- o A comparison table or short report explaining the differences.

Prompt: Compare two Python programs: one that calculates factorial without using functions and one that uses a user-defined factorial function. Explain the differences in terms of logic, reusability, and execution, and provide a short report.

Code:

Without Functions:

```
S-10.4 > ass1.1 > task-4 > ...
num = int(input("Enter a non-negative integer: "))
factorial = 1
if num < 0:
    print("Factorial does not exist for negative numbers.")
else:
    for i in range(2, num + 1):
        factorial *= i
    print(f"The factorial of {num} is {factorial}")
```

With Functions:

```
Tabnine | Edit | Test | Explain | Document
def factorial(n):
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

num = int(input("Enter a non-negative integer: "))
if num < 0:
    print("Factorial does not exist for negative numbers.")
else:
    print(f"The factorial of {num} is {factorial(num)}")
```

Output:

```
PS C:\Users\sonti\OneDrive\Documents\aiass2-  
sers\sonti\OneDrive\Documents\aiass2-9.2\AI  
Enter a non-negative integer: 6  
The factorial of 6 is 720  
PS C:\Users\sonti\OneDrive\Documents\aiass2-
```

Task 5: Iterative vs Recursive Factorial

Description:

Prompt GitHub Copilot to generate both iterative and recursive versions of the factorial function.

- Expected Output:
 - o Two correct implementations.
 - o A documented comparison of logic, performance, and execution flow between iterative and recursive approaches.

Prompt: Write two Python programs that calculate the factorial of a non-negative integer: one using an iterative approach with a loop, and one using recursion. Include input validation and comments explaining the logic. Provide a comparison of the two approaches in terms of logic, performance, and execution flow.

Code:

Iterative Factorial:

```
10.77 ass1.1 > task-5 > factorial_iterative
# Iterative factorial function
Tabnine | Edit | Test | Explain | Document
def factorial_iterative(n):
    """
    Calculate factorial iteratively.
    Parameters:
    n (int): Non-negative integer
    Returns:
    int: Factorial of n
    """
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Main program
num = int(input("Enter a non-negative integer: "))
if num < 0:
    print("Factorial does not exist for negative numbers.")
else:
    print(f"Iterative: The factorial of {num} is {factorial_iterative(num)}")
```

Recursive Factorial:

```
0.4 > ass1.1 > task-5 > ...
# Recursive factorial function
Tabnine | Edit | Test | Explain | Document
def factorial_recursive(n):
    """
    Calculate factorial recursively.
    Parameters:
    n (int): Non-negative integer

    Returns:
    int: Factorial of n
    """
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial_recursive(n - 1)

# Main program
if num >= 0:
    print(f"Recursive: The factorial of {num} is {factorial_recursive(num)}")
```


Output:

```
sers/sonti/OneDrive/Documents/aiass2-9.2/AI ASS-10  
Enter a non-negative integer: 4  
Iterative: The factorial of 4 is 24  
PS C:\Users\sonti\OneDrive\Documents\aiass2-9.2>
```