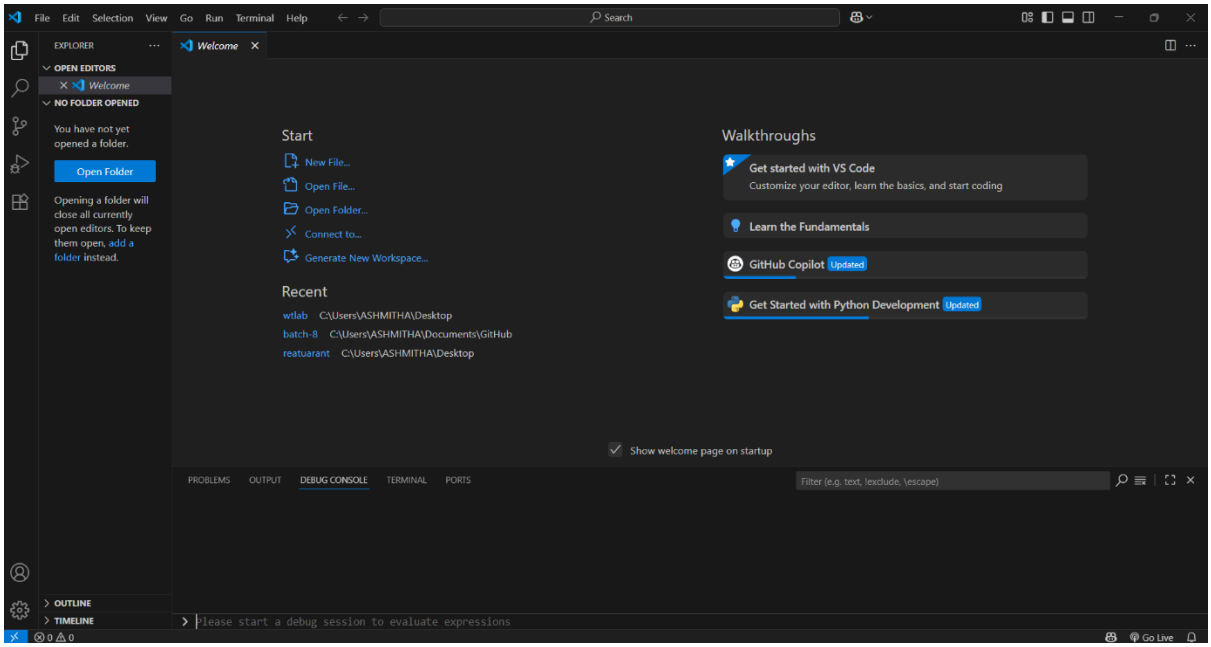
Assignment-1.1

Name: K. Shiva Shankar

Id:2403a510e5

#Task 0

● Install and configure GitHub Copilot in VS Code. Take screenshots .



#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 loopsdirectly in the main code).

● Expected Output:

o A working program that correctly calculates the factorial for user-provided input. Screenshots of the code generation process

Code:

```python
number = int(input("Enter a number: "))
factorial = 1

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

Output:

```
Enter a number: 8
The factorial of 8 is 40320
```

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

Code:

Improved Version

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

if number < 0:
    print("Factorial is not defined for negative numbers.")
elif number == 0:
    print("The factorial of 0 is 1")
else:
    factorial = 1
    for i in range(2, number + 1):
        factorial *= i
    print(f"The factorial of {number} is {factorial}")
```

Explanation of Improvements

- **Removed unnecessary multiplication:** The loop now starts from 2 instead of 1, since multiplying by 1 does not change the result.
- **Special case for 0:** Added an explicit check for `number == 0` to print the result directly, avoiding unnecessary looping.
- **Variable initialization:** The `factorial` variable is only initialized when needed (i.e., for positive numbers).
- **Performance:** These changes slightly reduce the number of loop iterations and unnecessary operations, making the code marginally more efficient and clear.

Output:

```
Enter a number: 5
The factorial of 5 is 120
```

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

Code:

```python
def factorial(n):
    if n < 0:
        return None  # Factorial not defined for negative numbers
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

number = int(input("Enter a number: "))
fact = factorial(number)
if fact is None:
    print("Factorial is not defined for negative numbers.")
else:
    print(f"The factorial of {number} is {fact}")
```

Output:

```
Enter a number: 0
The factorial of 0 is 1
```

Steps Copilot Followed to Generate the Function

1. **Defined a function** called `factorial` that takes an integer `n` as input.
2. **Checked for negative input** and returned `None` if `n` is negative, since factorial is not defined for negative numbers.
3. **Initialized a result variable** to 1 to store the factorial value.
4. **Used a loop** from 2 to `n` (inclusive) to multiply the result by each number in the range.
5. **Returned the result** after the loop completes.
6. **Took user input** and called the `factorial` function.
7. **Handled the output**: Printed an error message for negative input, or printed the factorial for valid input.

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.

Aspect	Without Function (Direct Code)	With Function (User-Defined Function)
Code	```python	
number = int(input("Enter a number: "))		
factorial = 1		
if number < 0:		

```
print("Factorial is not defined for negative numbers.")
```

```
else: for i in range(1, number + 1): factorial *= i print(f"The factorial of {number} is {factorial}") | python def factorial(n): if n < 0: return None
result = 1 for i in range(2, n + 1): result *= i return result

number = int(input("Enter a number: ")) fact = factorial(number) if fact is None: print("Factorial is not defined for negative numbers.") else:
print(f"The factorial of {number} is {fact}")
```

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

Code:

```python
# Iterative version
def factorial_iterative(n):
    if n < 0:
        return None
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Recursive version
def factorial_recursive(n):
    if n < 0:
        return None
    if n == 0 or n == 1:
        return 1
    return n * factorial_recursive(n - 1)

# Example usage
number = int(input("Enter a number: "))

fact_iter = factorial_iterative(number)
fact_rec = factorial_recursive(number)

if fact_iter is None or fact_rec is None:
    print("Factorial is not defined for negative numbers.")
else:
    print(f"Iterative: The factorial of {number} is {fact_iter}")
    print(f"Recursive: The factorial of {number} is {fact_rec}")
```

Output:

```
Enter a number: 5
Iterative: The factorial of 5 is 120
Recursive: The factorial of 5 is 120
```