

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

Use GitHub Copilot to generate a Python program that computes a mathematical product-based value directly in the main execution flow, without using any user-defined functions.

Constraint:

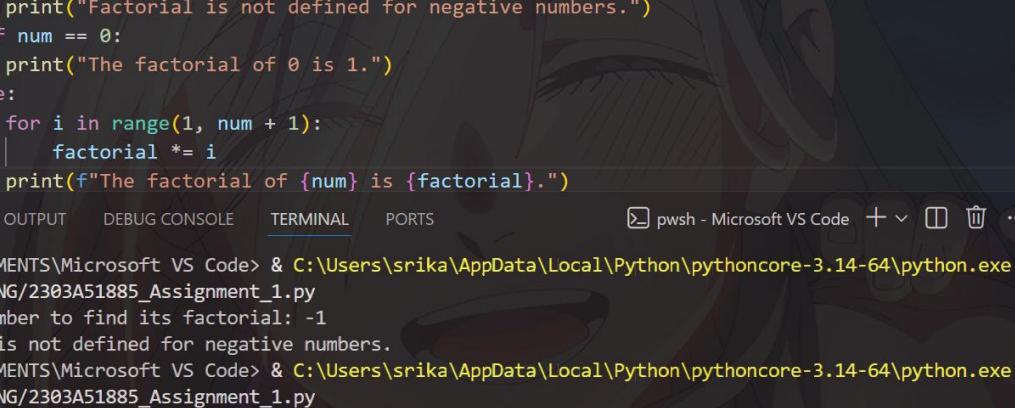
Do not define any custom function

Logic must be implemented using loops and variables only

Copilot suggestions:

```
#Write a facotrial code in python without functions
num = int(input("Enter a number to find its factorial: "))
factorial = 1
if num < 0:
    print("Factorial is not defined 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}.")
```

Code:



```
2 num = int(input("Enter a number to find its factorial: "))
3 factorial = 1
4 if num < 0:
5     print("Factorial is not defined for negative numbers.")
6 elif num == 0:
7     print("The factorial of 0 is 1.")
8 else:
9     for i in range(1, num + 1):
10        factorial *= i
11    print(f"The factorial of {num} is {factorial}.")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS pwsh - Microsoft VS Code + ⌂ ⌂ ⌂ ... [] X

```
PS C:\DOCUMENTS\Microsoft VS Code> & C:\Users\srika\AppData\Local\Python\pythoncore-3.14-64\python.exe e:/AI_ASSISTED_CODING/2303A51885_Assignment_1.py
Enter a number to find its factorial: -1
Factorial is not defined for negative numbers.
PS C:\DOCUMENTS\Microsoft VS Code> & C:\Users\srika\AppData\Local\Python\pythoncore-3.14-64\python.exe e:/AI_ASSISTED_CODING/2303A51885_Assignment_1.py
Enter a number to find its factorial: 0
The factorial of 0 is 1.
PS C:\DOCUMENTS\Microsoft VS Code> & C:\Users\srika\AppData\Local\Python\pythoncore-3.14-64\python.exe e:/AI_ASSISTED_CODING/2303A51885_Assignment_1.py
Enter a number to find its factorial: 5
The factorial of 5 is 120.
```

Reflection:

So when I given the prompt “#Write a factorial code in python without functions “

Copilot helped me to write the code smoothly and efficiently. When I written the prompt in comments in the first line. It has generated code within a few seconds. But initially the code is not very optimal because initially when we give a basic prompt it gives basic code like it will not handle any other cases like in the above basic code it didn't give any functions and it didn't code how to handle about float values and strings. For a beginner starting with copilot is a bad idea in my opinion because he doesn't have any idea about coding. If a person who knows basic coding and if he stuck in the middle of code. He can ask Copilot to complete the code.

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

Task Description

Analyze the code generated in Task 1 and use Copilot again to:

- Reduce unnecessary variables
- Improve loop clarity
- Enhance readability and efficiency

Copilot suggestion

```
#Write a facotrail code in python without functions simplify logic and make it more readable
number = int(input("Enter a number to find its factorial: "))
factorial_result = 1
if number < 0: # Check for negative input
    print("Factorial is not defined for negative numbers.")
elif number == 0: # Factorial of 0 is 1
    print("The factorial of 0 is 1.")
else: # Calculate factorial for positive numbers
    for i in range(1, number + 1): # Loop from 1 to the number
        factorial_result *= i
    print(f"The factorial of {number} is {factorial_result}.")
```

Code:

```
#Write a facotrial code in python without functions simplify logic and make it more readable
"""
docstring: This code calculates the factorial of a given non-negative integer.
"""

number = int(input("Enter a number to find its factorial: "))
factorial_result = 1
if number < 0: # Check for negative input
    print("Factorial is not defined for negative numbers.")
elif number == 0: # Factorial of 0 is 1
    print("The factorial of 0 is 1.")
else: # Calculate factorial for positive numbers
    for i in range(1, number + 1): # Loop from 1 to the number
        factorial_result *= i # Multiply to get factorial
    print(f"The factorial of {number} is {factorial_result}.") # Display the result
```

What is improved:

So if we observe the new code when the prompt is given as “**#Write a factorial code in python without functions simplify logic and make it more readable**”.

Copilot has given a code which contains the docstring. It is like a description of code. By this we will be able to understand what the entire code will be doing. As we can observe that in the code we can observe that some important lines have comments which tell us that which part of code is doing what, like what the if conditions will do, what the loops will do and what the statement in the loop does. It makes the code more readable and more understandable even for a person who is from a non technical background to understand while reading it.

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

❖ Scenario

The same logic now needs to be reused in multiple scripts

Constraints

- Use meaningful function and variable names
- Include inline comments (preferably suggested by Copilot)

Copilot suggestion:

```
#Write a factorial code in python with functions
def calculate_factorial(n):
    """Calculate the factorial of a non-negative integer n."""
    if n < 0:
        return "Factorial is not defined for negative numbers."
    elif n == 0:
        return 1
    else:
        factorial = 1
        for i in range(1, n + 1):
            factorial *= i
        return factorial
```

Code:

```
28 #Write a factorial code in python with functions
29 def calculate_factorial(n):
30     """Calculate the factorial of a non-negative integer n."""
31     if n < 0:
32         return "Factorial is not defined for negative numbers."
33     elif n == 0:
34         return 1
35     else:
36         factorial = 1
37         for i in range(1, n + 1):
38             factorial *= i
39         return factorial
40 print("Factorial Calculator")
41 print(calculate_factorial(0)) # Output: 1
42 print(calculate_factorial(5)) # Output: 120
43 print(calculate_factorial(-3)) # Output: Factorial is not defined for negative numbers.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

pwsh - Mic

```
Factorial Calculator
1
120
Factorial is not defined for negative numbers.
```

Description:

So if we observe the code when the new prompt is “#Write a factorial code in python with functions”

As we know that when we write a code it contains many parts in it . So to remove confusion and maintain a clarity we create a functions in the code because every function has its own work to do. For example if we see this function we can understand that it calculates the factorial of a given value. So when we write a multiple functions if a output of a function is wrong we can understand that there is a wrong in the function instead of checking the entire code we can just generally go and check inside the function . By functions we can increase the codes reusability .So functions makes our codes reusable and easy to understand.

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

Simple code

```
#Write a facotrail code in python without functions simplify logic and make it more readable
number = int(input("Enter a number to find its factorial: "))
factorial_result = 1
if number < 0: # Check for negative input
    print("Factorial is not defined for negative numbers.")
elif number == 0: # Factorial of 0 is 1
    print("The factorial of 0 is 1.")
else: # Calculate factorial for positive numbers
    for i in range(1, number + 1): # Loop from 1 to the number
        factorial_result *= i
    print(f"The factorial of {number} is {factorial_result}.")
```

Functions code

```
#Write a facotrail code in python with functions
def calculate_factorial(n):
    """Calculate the factorial of a non-negative integer n."""
    if n < 0:
        return "Factorial is not defined for negative numbers."
    elif n == 0:
        return 1
    else:
        factorial = 1
        for i in range(1, n + 1):
            factorial *= i
        return factorial
```

COMPARISON	CODE WITH NO FUNCTIONS	CODE WITH FUNCTIONS
Readability	It is mixed with the total code .So it is hard to read	It is written in a module by this the code is easy to read.
Understandability	The code is mixed totally so it is hard to understand the code.	We can create a docstring by this the code is easily readable and understandable.
Debugging	It is very hard to debug the code.	It Is easy to debug the code.
Portray	The person who reads the code thinks it is written by a beginner.	The person feels coder is not at beginner level.
Reusability	It is not reusable	It is reusable/

Task 5: AI-Generated Iterative vs Recursive Thinking

Your mentor wants to test how well AI understands different computational paradigms

Non-Recursive code

```

28  #Write a facotrial code in python with functions
29  def calculate_factorial(n):
30      """Calculate the factorial of a non-negative integer n."""
31      if n < 0:
32          return "Factorial is not defined for negative numbers."
33      elif n == 0:
34          return 1
35      else:
36          factorial = 1
37          for i in range(1, n + 1):
38              factorial *= i
39          return factorial
40  print("Factorial Calculator")
41  print(calculate_factorial(0)) # Output: 1
42  print(calculate_factorial(5)) # Output: 120
43  print(calculate_factorial(-3)) # Output: Factorial is not defined for negative numbers.

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

pws - Mic

```

Factorial Calculator
1
120
Factorial is not defined for negative numbers.

```

Recursion Code

```
45  #write a facotrial code in python with recursion
46  def factorial_recursive(n):
47      """Calculate the factorial of a non-negative integer n using recursion."""
48      if n < 0:
49          return "Factorial is not defined for negative numbers."
50      elif n == 0:
51          return 1
52      else:
53          return n * factorial_recursive(n - 1)
54  print("Factorial Calculator using Recursion")
55  print(factorial_recursive(0)) # Output: 1
56  print(factorial_recursive(5)) # Output: 120
57  print(factorial_recursive(-3)) # Output: Factorial is not defined for negative numbers.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[] pwsh - M

```
Factorial Calculator using Recursion
1
120
Factorial is not defined for negative numbers.
```

Description

So for the both codes the prompts are

"#Write a factorial code in python with functions"

"#write a factorial code in python with recursion"

A code which contains loops can also be done recursion, recursion means repetition of a function . So for some codes, writing recursion is required but if we use recursion a lot it takes a lot of space. So for some codes functions are required and for some codes recursion is required. If we have large inputs we can break them and use recursions. A function may not use recursion but a recursion always requires function.