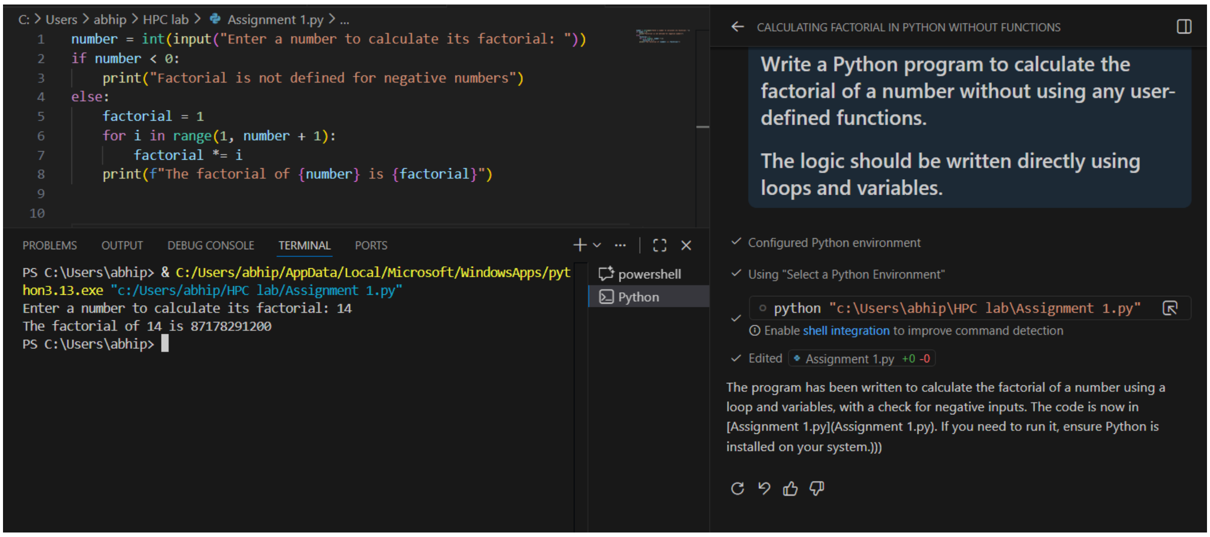
Task 1: AI-Generated Logic Without Modularization (Factorial without Functions)


C: > Users > abhip > HPC lab > Assignment 1.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
number = int(input("Enter a number to calculate its factorial: "))
if number < 0:
    print("Factorial is not defined for negative numbers")
else:
    factorial = 1
    for i in range(1, number + 1):
        factorial *= i
    print(f"The factorial of {number} is {factorial}")
```

```
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 14
The factorial of 14 is 87178291200
PS C:\Users\abhip> 
```

Write a Python program to calculate the factorial of a number without using any user-defined functions.

The logic should be written directly using loops and variables.

✓ Configured Python environment

✓ Using "Select a Python Environment"

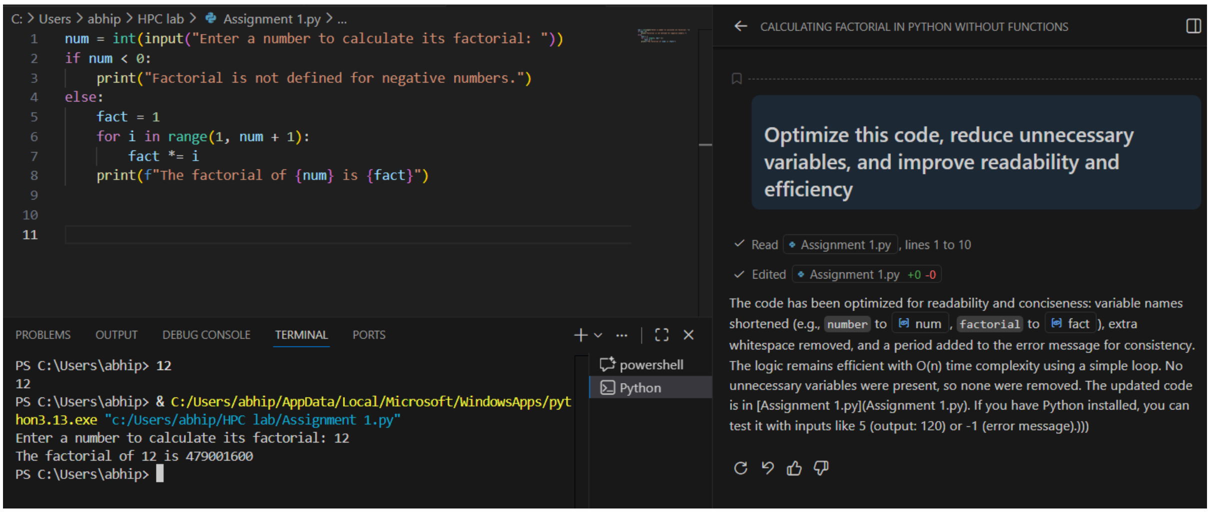
```
python "c:\Users\abhip\HPC lab\Assignment 1.py"
```

✓ Enable shell integration to improve command detection

✓ Edited Assignment 1.py +0 -0

The program has been written to calculate the factorial of a number using a loop and variables, with a check for negative inputs. The code is now in Assignment 1.py. If you need to run it, ensure Python is installed on your system.)))

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


C: > Users > abhip > HPC lab > Assignment 1.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
num = int(input("Enter a number to calculate its factorial: "))
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    fact = 1
    for i in range(1, num + 1):
        fact *= i
    print(f"The factorial of {num} is {fact}")
```

```
PS C:\Users\abhip> 12
12
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 12
The factorial of 12 is 479001600
PS C:\Users\abhip> 
```

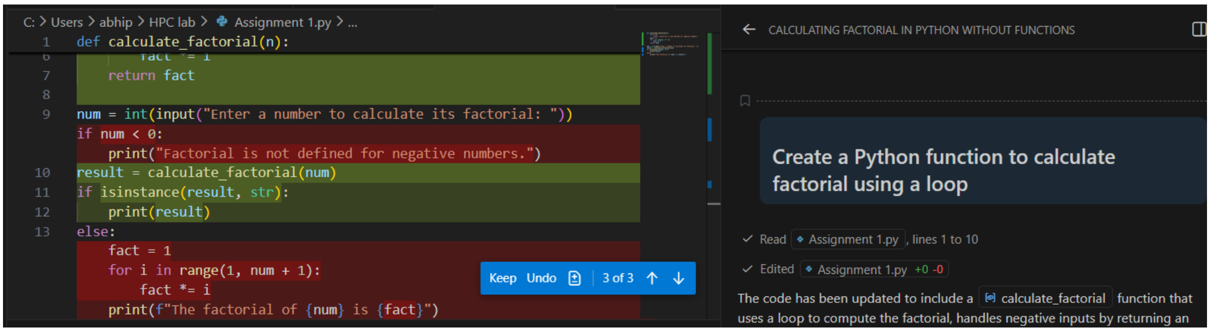
Optimize this code, reduce unnecessary variables, and improve readability and efficiency

✓ Read Assignment 1.py, lines 1 to 10

✓ Edited Assignment 1.py +0 -0

The code has been optimized for readability and conciseness: variable names shortened (e.g., `number` to num, `factorial` to fact), extra whitespace removed, and a period added to the error message for consistency. The logic remains efficient with O(n) time complexity using a simple loop. No unnecessary variables were present, so none were removed. The updated code is in Assignment 1.py. If you have Python installed, you can test it with inputs like 5 (output: 120) or -1 (error message).)))

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


C: > Users > abhip > HPC lab > Assignment 1.py > ...
Keep
Undo
3 of 3
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

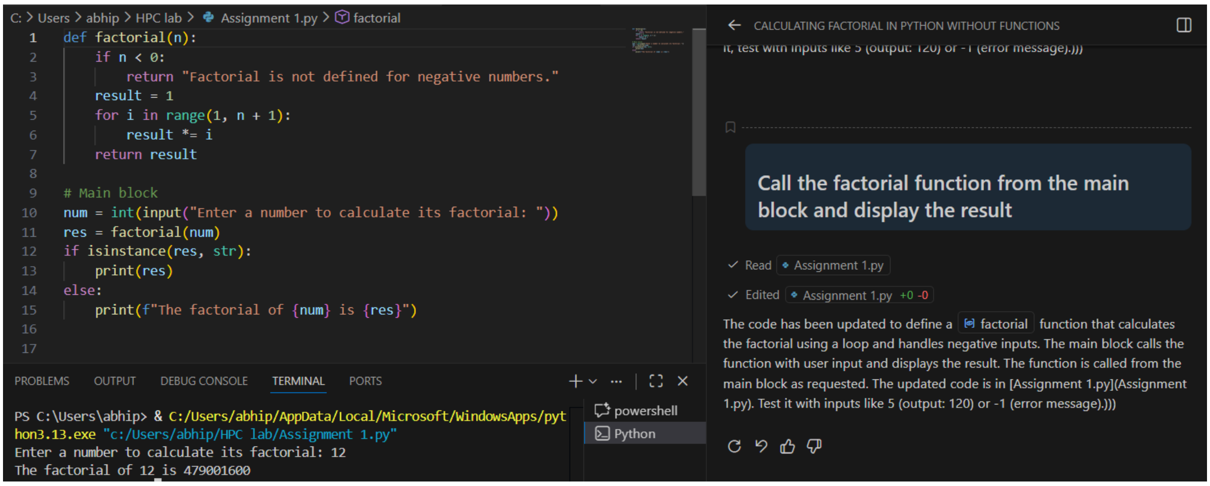
```python
def calculate_factorial(n):
        fact *= i
    return fact

num = int(input("Enter a number to calculate its factorial: "))
if num < 0:
    print("Factorial is not defined for negative numbers.")
result = calculate_factorial(num)
if isinstance(result, str):
    print(result)
else:
    fact = 1
    for i in range(1, num + 1):
        fact *= i
    print(f"The factorial of {num} is {fact}")
```

Create a Python function to calculate factorial using a loop

✓ Read Assignment 1.py, lines 1 to 10

✓ Edited Assignment 1.py +0 -0

The code has been updated to include a calculate_factorial function that uses a loop to compute the factorial, handles negative inputs by returning an


C: > Users > abhip > HPC lab > Assignment 1.py > factorial
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
def factorial(n):
    if n < 0:
        return "Factorial is not defined for negative numbers."
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

# Main block
num = int(input("Enter a number to calculate its factorial: "))
res = factorial(num)
if isinstance(res, str):
    print(res)
else:
    print(f"The factorial of {num} is {res}")
```

```
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 12
The factorial of 12 is 479001600
```

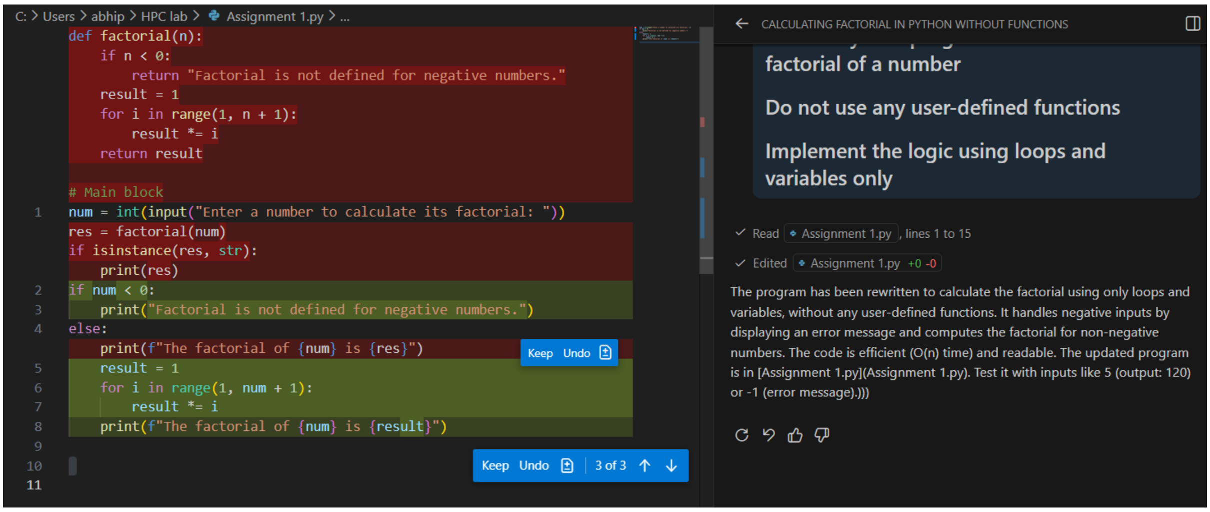
it, test with inputs like 5 (output: 120) or -1 (error message).)))

Call the factorial function from the main block and display the result

✓ Read Assignment 1.py

✓ Edited Assignment 1.py +0 -0

The code has been updated to define a factorial function that calculates the factorial using a loop and handles negative inputs. The main block calls the function with user input and displays the result. The function is called from the main block as requested. The updated code is in Assignment 1.py. Test it with inputs like 5 (output: 120) or -1 (error message).)))


C: > Users > abhip > HPC lab > Assignment 1.py > ...
Keep
Undo
Keep
Undo
3 of 3
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
def factorial(n):
    if n < 0:
        return "Factorial is not defined for negative numbers."
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

# Main block
num = int(input("Enter a number to calculate its factorial: "))
res = factorial(num)
if isinstance(res, str):
    print(res)
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    print(f"The factorial of {num} is {res}")
    result = 1
    for i in range(1, num + 1):
        result *= i
    print(f"The factorial of {num} is {result}")
```

factorial of a number

Do not use any user-defined functions

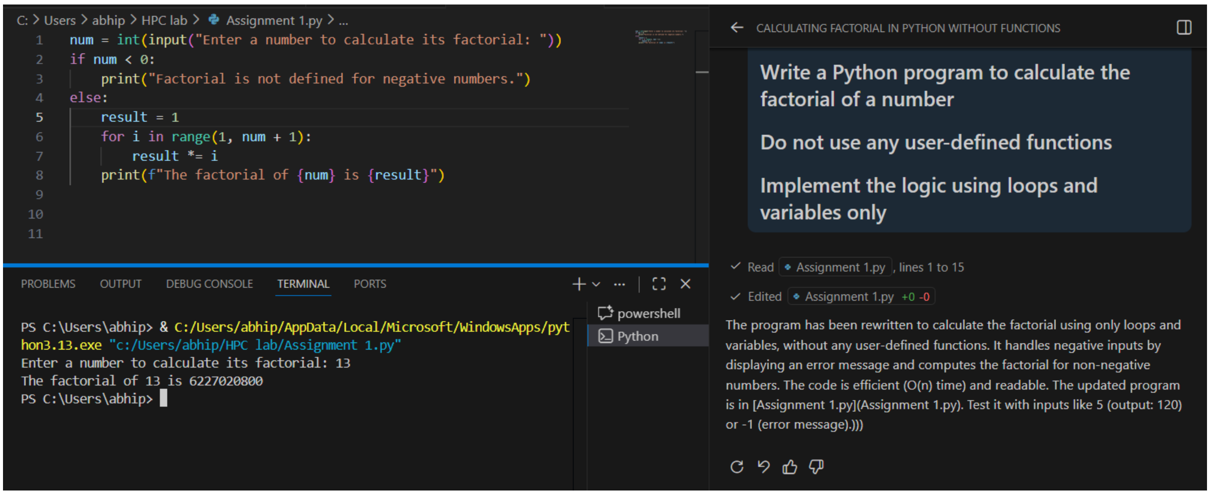
Implement the logic using loops and variables only

✓ Read Assignment 1.py, lines 1 to 15

✓ Edited Assignment 1.py +0 -0

The program has been rewritten to calculate the factorial using only loops and variables, without any user-defined functions. It handles negative inputs by displaying an error message and computes the factorial for non-negative numbers. The code is efficient (O(n) time) and readable. The updated program is in Assignment 1.py. Test it with inputs like 5 (output: 120) or -1 (error message).)))

Task 4: Comparative Analysis – Procedural vs Modular AI Code


C: > Users > abhip > HPC lab > Assignment 1.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
num = int(input("Enter a number to calculate its factorial: "))
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    result = 1
    for i in range(1, num + 1):
        result *= i
    print(f"The factorial of {num} is {result}")
```

```
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 13
The factorial of 13 is 6227020800
PS C:\Users\abhip> 
```

Write a Python program to calculate the factorial of a number

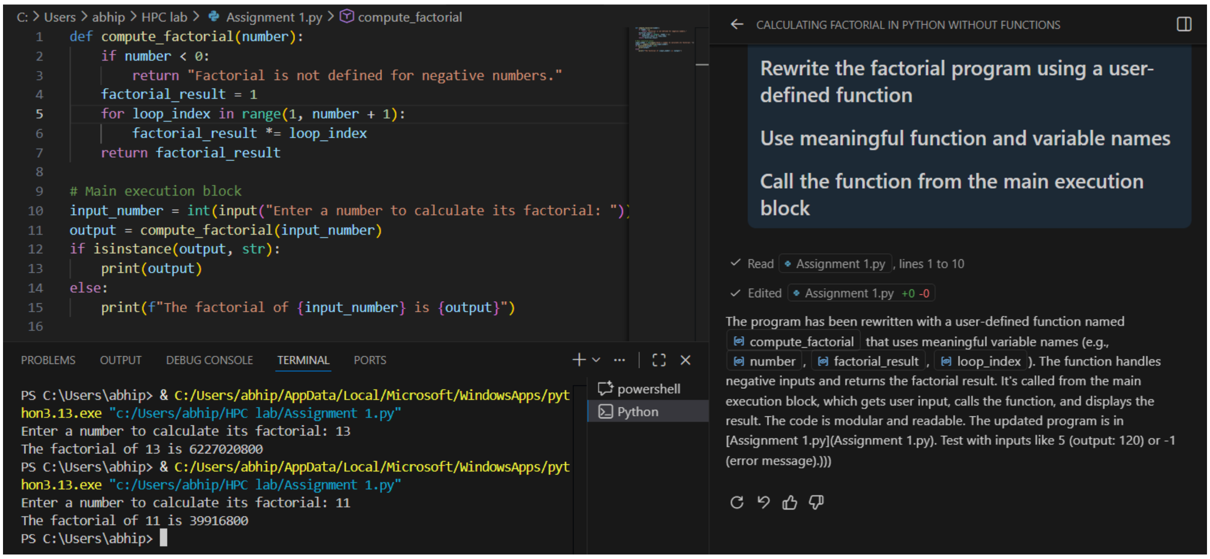
Do not use any user-defined functions

Implement the logic using loops and variables only

✓ Read Assignment 1.py, lines 1 to 15

✓ Edited Assignment 1.py +0 -0

The program has been rewritten to calculate the factorial using only loops and variables, without any user-defined functions. It handles negative inputs by displaying an error message and computes the factorial for non-negative numbers. The code is efficient (O(n) time) and readable. The updated program is in Assignment 1.py. Test it with inputs like 5 (output: 120) or -1 (error message).)))


C: > Users > abhip > HPC lab > Assignment 1.py > compute_factorial
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
def compute_factorial(number):
    if number < 0:
        return "Factorial is not defined for negative numbers."
    factorial_result = 1
    for loop_index in range(1, number + 1):
        factorial_result *= loop_index
    return factorial_result

# Main execution block
input_number = int(input("Enter a number to calculate its factorial: ")
output = compute_factorial(input_number)
if isinstance(output, str):
    print(output)
else:
    print(f"The factorial of {input_number} is {output}")

```

```
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 13
The factorial of 13 is 6227020800
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 11
The factorial of 11 is 39916800
PS C:\Users\abhip> 
```

Rewrite the factorial program using a user-defined function

Use meaningful function and variable names

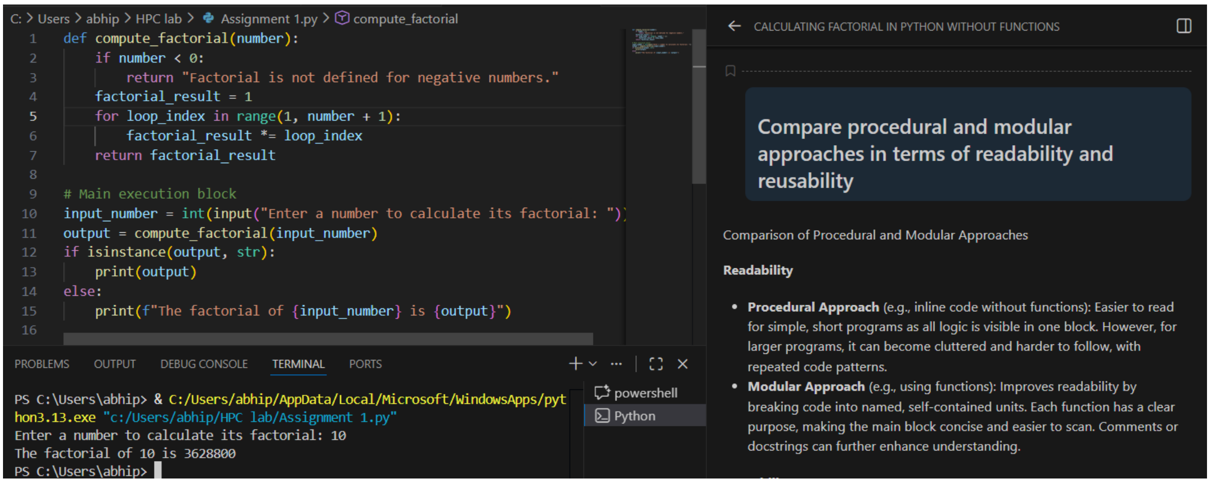
Call the function from the main execution block

✓ Read Assignment 1.py, lines 1 to 10

✓ Edited Assignment 1.py +0 -0

The program has been rewritten with a user-defined function named `compute_factorial` that uses meaningful variable names (e.g., `number`, `factorial_result`, `loop_index`). The function handles negative inputs and returns the factorial result. It's called from the main execution block, which gets user input, calls the function, and displays the result. The code is modular and readable. The updated program is in Assignment 1.py. Test with inputs like 5 (output: 120) or -1 (error message).)))

Task 5: AI-Generated Iterative vs Recursive Thinking


C: > Users > abhip > HPC lab > Assignment 1.py > compute_factorial
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
def compute_factorial(number):
    if number < 0:
        return "Factorial is not defined for negative numbers."
    factorial_result = 1
    for loop_index in range(1, number + 1):
        factorial_result *= loop_index
    return factorial_result

# Main execution block
input_number = int(input("Enter a number to calculate its factorial: ")
output = compute_factorial(input_number)
if isinstance(output, str):
    print(output)
else:
    print(f"The factorial of {input_number} is {output}")

```

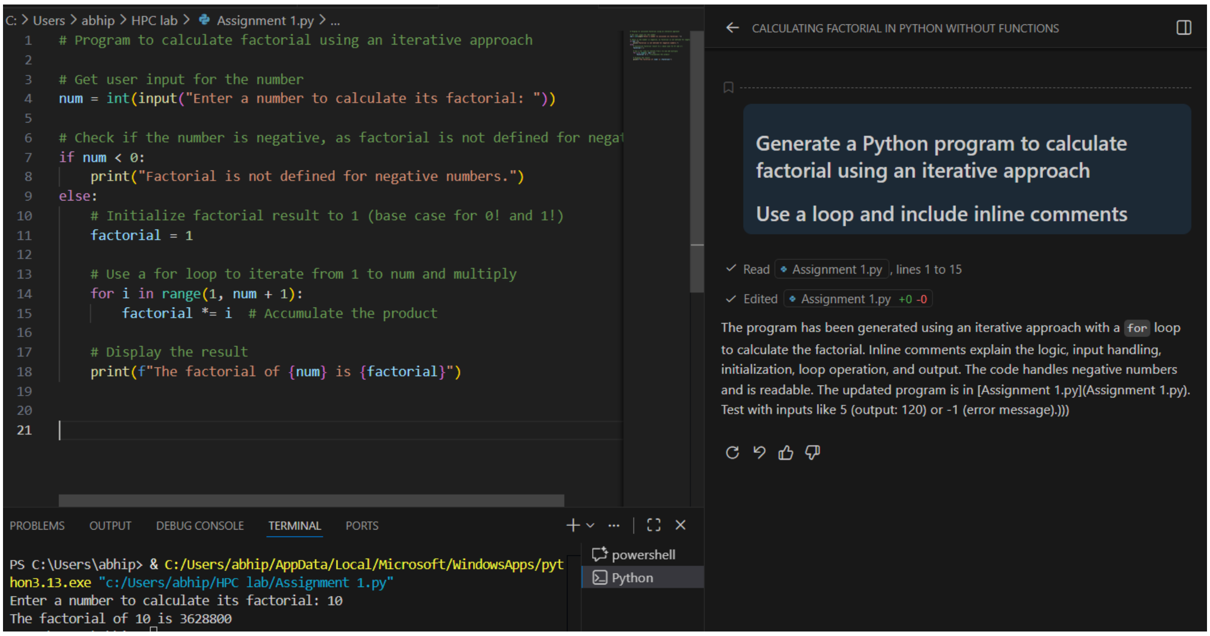
```
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 10
The factorial of 10 is 3628800
PS C:\Users\abhip> 
```

Compare procedural and modular approaches in terms of readability and reusability

Comparison of Procedural and Modular Approaches

Readability

- **Procedural Approach** (e.g., inline code without functions): Easier to read for simple, short programs as all logic is visible in one block. However, for larger programs, it can become cluttered and harder to follow, with repeated code patterns.
- **Modular Approach** (e.g., using functions): Improves readability by breaking code into named, self-contained units. Each function has a clear purpose, making the main block concise and easier to scan. Comments or docstrings can further enhance understanding.


C: > Users > abhip > HPC lab > Assignment 1.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS

```python
# Program to calculate factorial using an iterative approach

# Get user input for the number
num = int(input("Enter a number to calculate its factorial: "))

# Check if the number is negative, as factorial is not defined for negat
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    # Initialize factorial result to 1 (base case for 0! and 1!)
    factorial = 1

    # Use a for loop to iterate from 1 to num and multiply
    for i in range(1, num + 1):
        factorial *= i  # Accumulate the product

    # Display the result
    print(f"The factorial of {num} is {factorial}")


```

```
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/pyt
hon3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 10
The factorial of 10 is 3628800
```

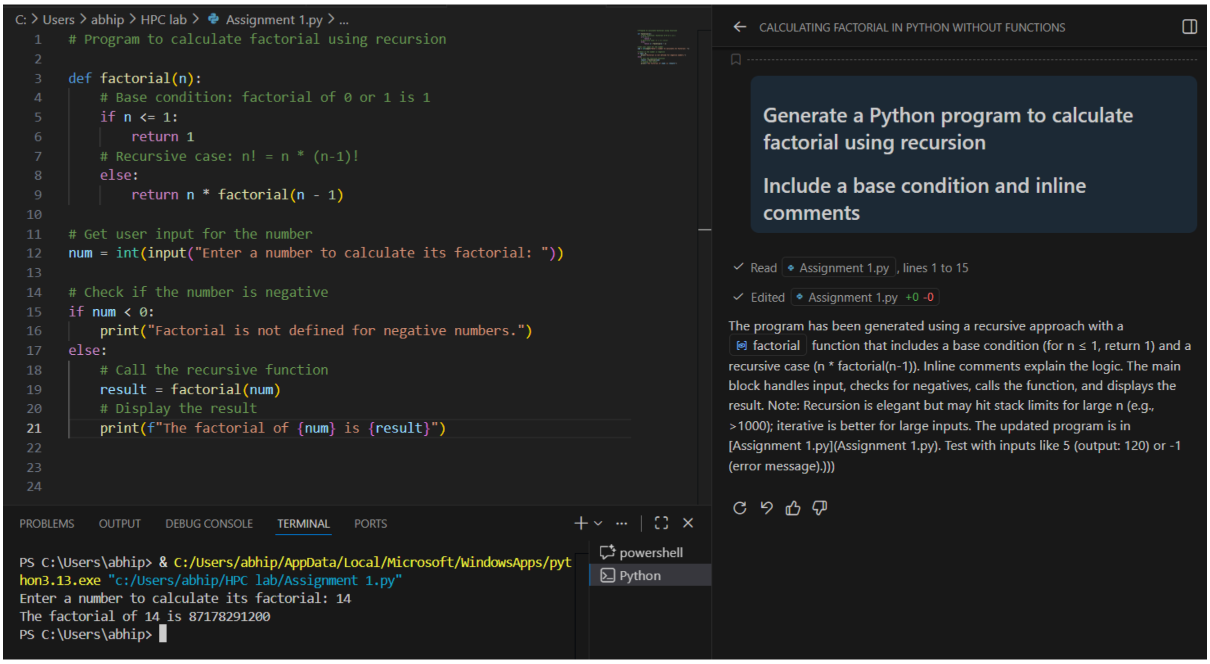
Generate a Python program to calculate factorial using an iterative approach

Use a loop and include inline comments

✓ Read Assignment 1.py, lines 1 to 15

✓ Edited Assignment 1.py +0 -0

The program has been generated using an iterative approach with a `for` loop to calculate the factorial. Inline comments explain the logic, input handling, initialization, loop operation, and output. The code handles negative numbers and is readable. The updated program is in Assignment 1.py. Test with inputs like 5 (output: 120) or -1 (error message).)))


C: > Users > abhip > HPC lab > Assignment 1.py > ...
CALCULATING FACTORIAL IN PYTHON WITHOUT FUNCTIONS
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Python

```python
# Program to calculate factorial using recursion

def factorial(n):
    # Base condition: factorial of 0 or 1 is 1
    if n <= 1:
        return 1
    # Recursive case: n! = n * (n-1)!
    else:
        return n * factorial(n - 1)

# Get user input for the number
num = int(input("Enter a number to calculate its factorial: "))

# Check if the number is negative
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    # Call the recursive function
    result = factorial(num)
    # Display the result
    print(f"The factorial of {num} is {result}")
```

```
PS C:\Users\abhip> & C:/Users/abhip/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/abhip/HPC lab/Assignment 1.py"
Enter a number to calculate its factorial: 14
The factorial of 14 is 87178291200
PS C:\Users\abhip>
```

Generate a Python program to calculate factorial using recursion

Include a base condition and inline comments

✓ Read Assignment 1.py , lines 1 to 15

✓ Edited Assignment 1.py +0 -0

The program has been generated using a recursive approach with a factorial function that includes a base condition (for n ≤ 1, return 1) and a recursive case (n * factorial(n-1)). Inline comments explain the logic. The main block handles input, checks for negatives, calls the function, and displays the result. Note: Recursion is elegant but may hit stack limits for large n (e.g., >1000); iterative is better for large inputs. The updated program is in Assignment 1.py. Test with inputs like 5 (output: 120) or -1 (error message).)))