

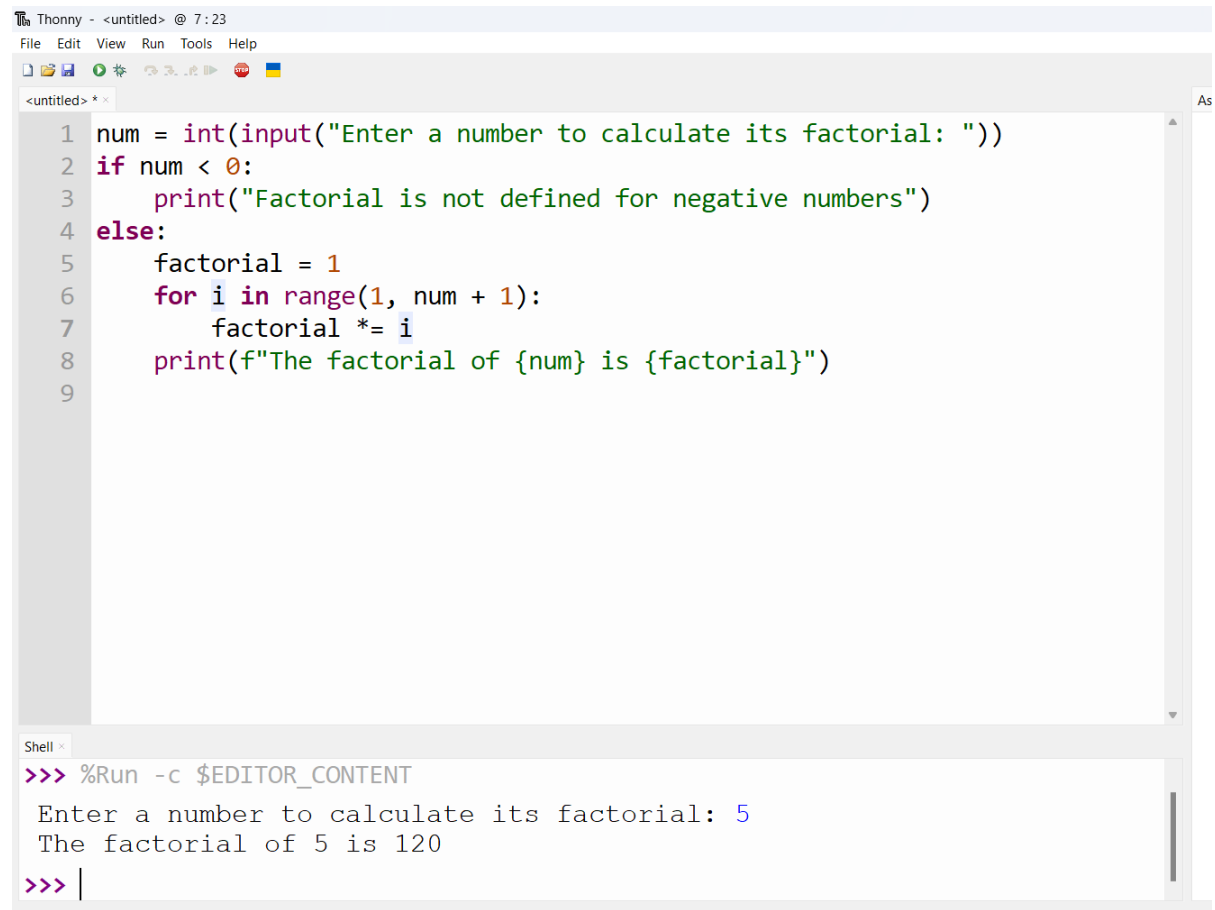
ASSIGNMENT-1

NAME:-K Ujwal

BATCH-16

HTNO:-2303A51058

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

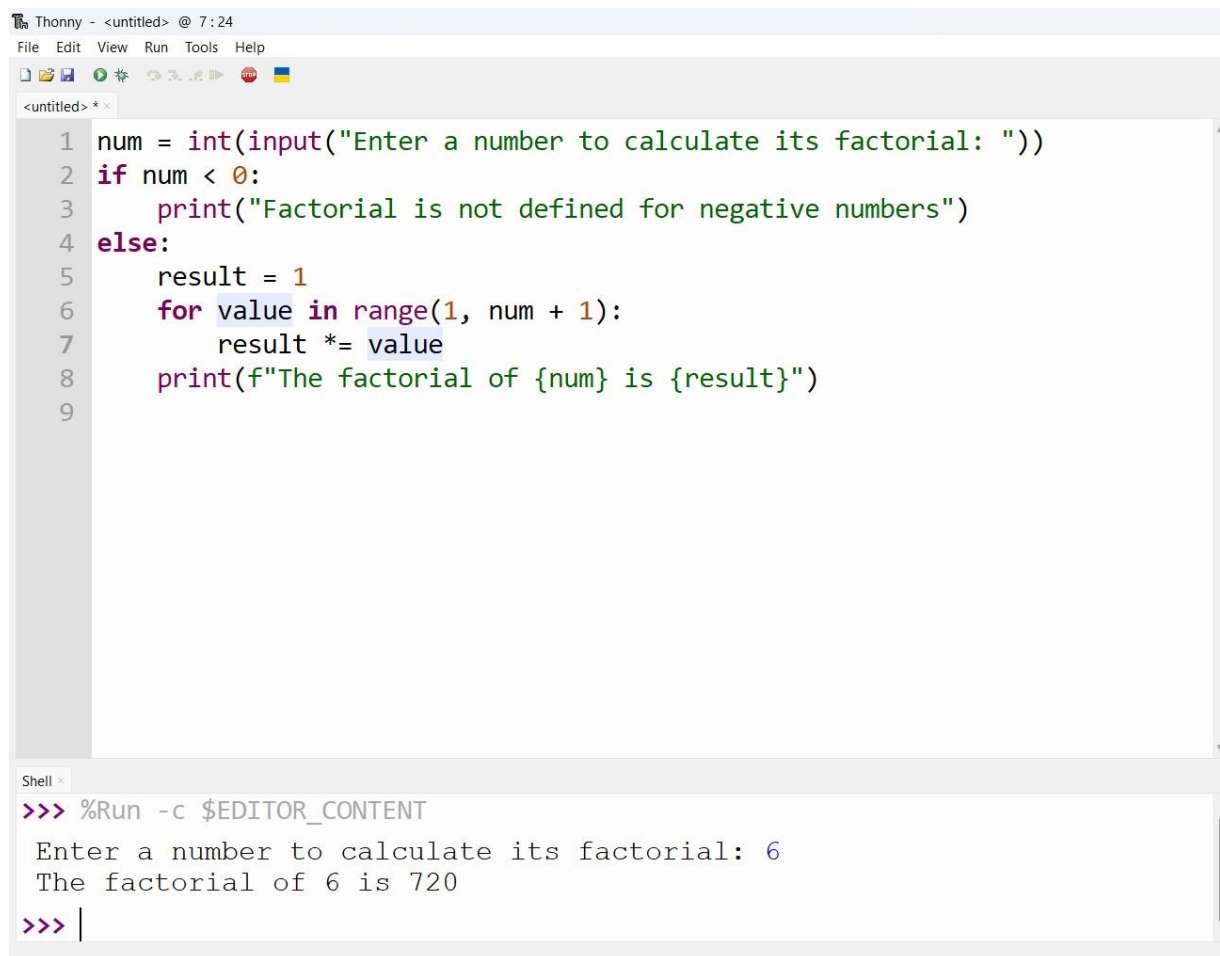


```
Thonny - <untitled> @ 7:23
File Edit View Run Tools Help

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

Shell
>>> %Run -c $EDITOR_CONTENT
Enter a number to calculate its factorial: 5
The factorial of 5 is 120
>>> |
```

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



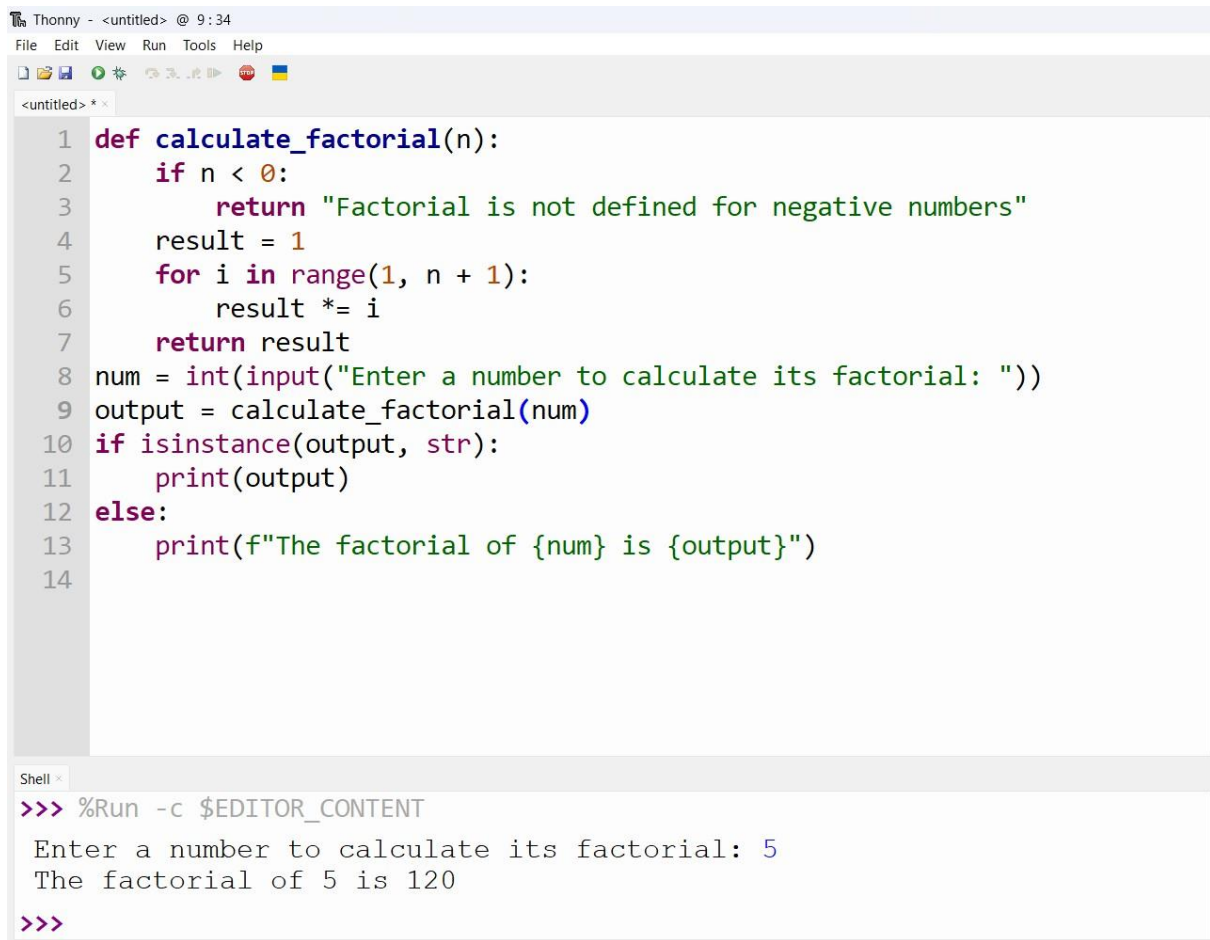
The image shows the Thonny IDE interface. The top menu bar includes File, Edit, View, Run, Tools, and Help. Below the menu is a toolbar with icons for file operations and execution. The main editor window, titled '<untitled> *', contains the following Python code:

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

Below the editor is a shell window titled 'Shell'. It shows the command `>>> %Run -c $EDITOR_CONTENT` being executed. The output of the script is displayed in the shell:

```
>>> %Run -c $EDITOR_CONTENT
Enter a number to calculate its factorial: 6
The factorial of 6 is 720
>>> |
```

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



The screenshot shows the Thonny Python IDE interface. The top menu bar includes File, Edit, View, Run, Tools, and Help. Below the menu is a toolbar with icons for file operations and running code. The main editor window, titled '<untitled> *', contains the following Python code:

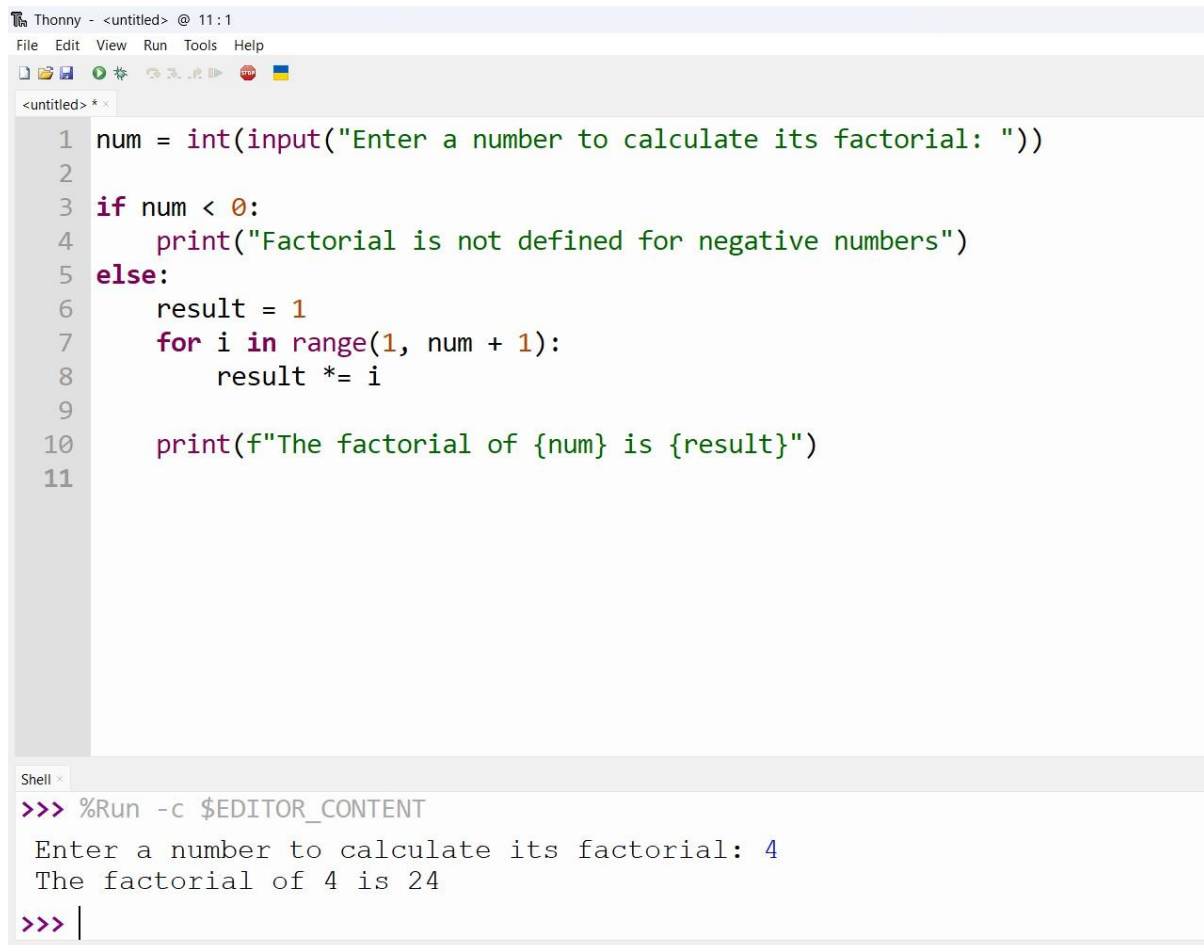
```
1 def calculate_factorial(n):
2     if n < 0:
3         return "Factorial is not defined for negative numbers"
4     result = 1
5     for i in range(1, n + 1):
6         result *= i
7     return result
8 num = int(input("Enter a number to calculate its factorial: "))
9 output = calculate_factorial(num)
10 if isinstance(output, str):
11     print(output)
12 else:
13     print(f"The factorial of {num} is {output}")
14
```

Below the editor is a Shell window titled 'Shell'. It shows the command prompt running the code:

```
>>> %Run -c $EDITOR_CONTENT
Enter a number to calculate its factorial: 5
The factorial of 5 is 120
>>>
```

Task 4: Comparative Analysis – Procedural vs Modular AI Code

Procedural (Without Function):-



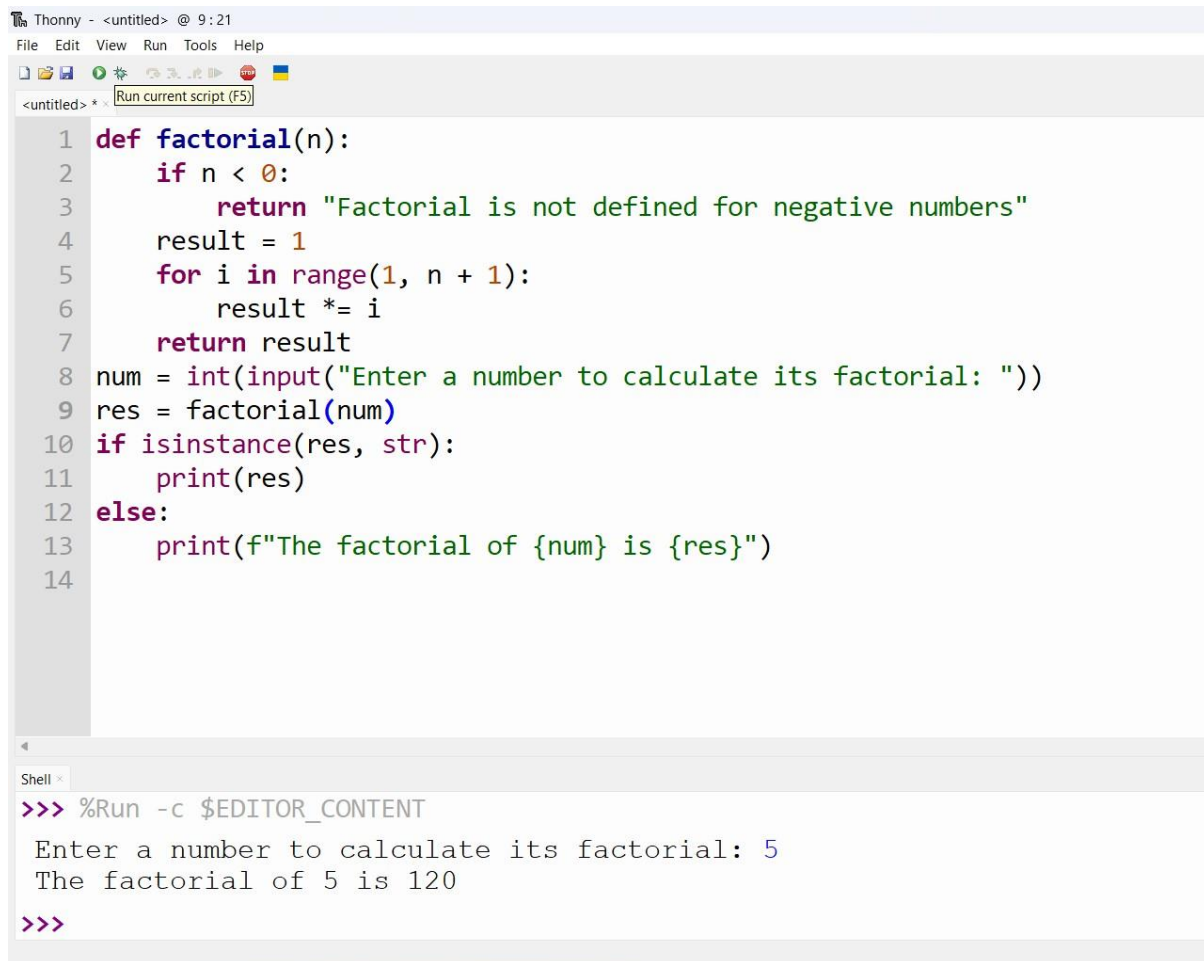
The image shows a screenshot of the Thonny Python IDE. The main window displays a Python script for calculating the factorial of a number. The script uses an if-else statement to handle negative numbers and a for loop to calculate the factorial for positive numbers. The script is as follows:

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

Below the script, the Shell window shows the execution output:

```
>>> %Run -c $EDITOR_CONTENT
Enter a number to calculate its factorial: 4
The factorial of 4 is 24
>>> |
```

Modular (With Function)



The screenshot shows the Thonny Python IDE interface. The top menu bar includes File, Edit, View, Run, Tools, and Help. Below the menu is a toolbar with icons for file operations, running, and debugging. The main editor window displays a Python script for calculating factorials. The script defines a function `factorial(n)` that returns an error message for negative numbers, initializes `result = 1`, and uses a `for` loop to calculate the factorial. It then prompts the user for a number, calls the function, and prints the result with a formatted string. The bottom panel shows a shell window where the script is executed, displaying the input '5' and the output 'The factorial of 5 is 120'.

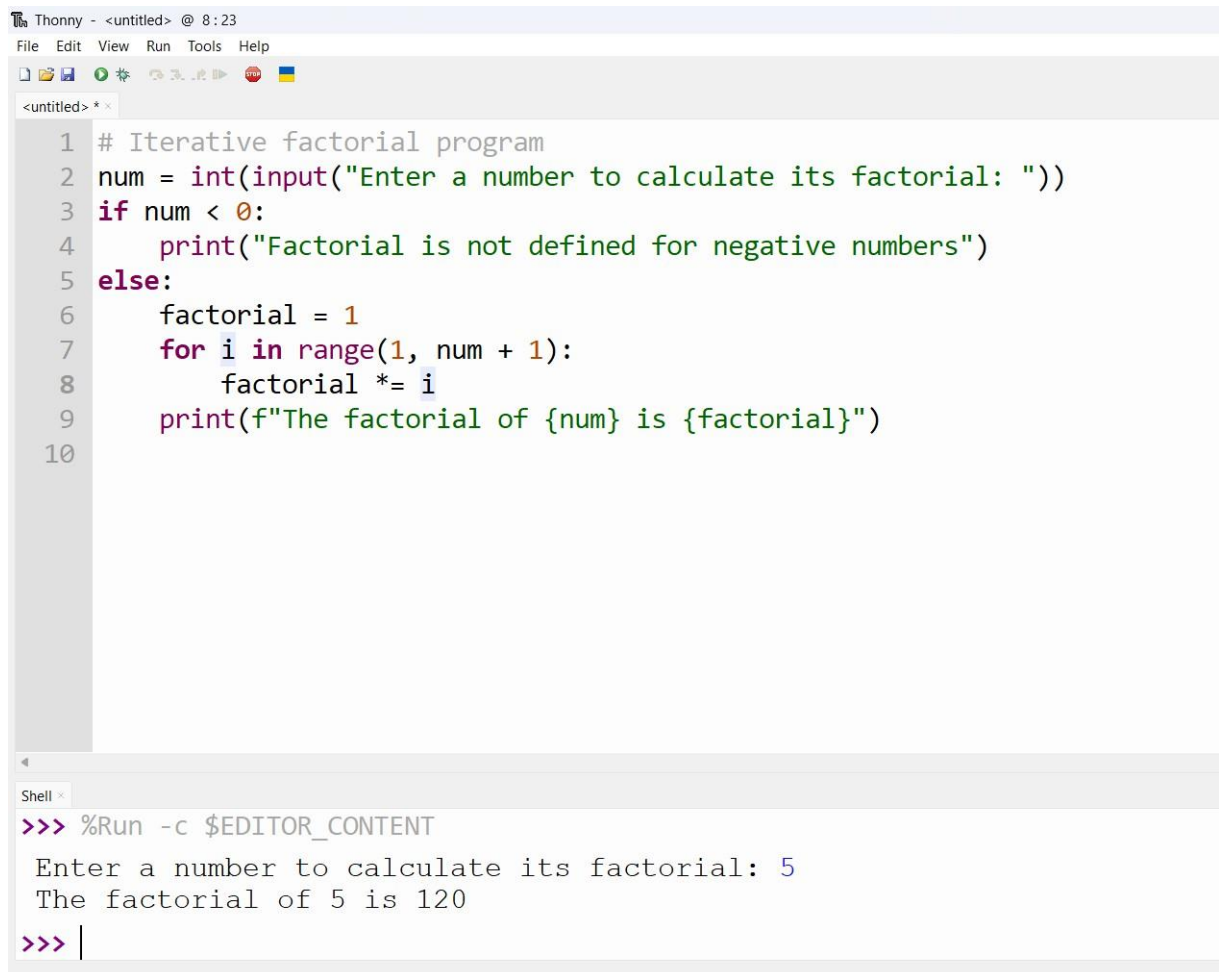
```
1 def factorial(n):
2     if n < 0:
3         return "Factorial is not defined for negative numbers"
4     result = 1
5     for i in range(1, n + 1):
6         result *= i
7     return result
8 num = int(input("Enter a number to calculate its factorial: "))
9 res = factorial(num)
10 if isinstance(res, str):
11     print(res)
12 else:
13     print(f"The factorial of {num} is {res}")
14
```

Shell x

```
>>> %Run -c $EDITOR_CONTENT
Enter a number to calculate its factorial: 5
The factorial of 5 is 120
>>>
```

Task 5: AI-Generated Iterative vs Recursive Thinking

Iterative Approach



```
Thonny - <untitled> @ 8:23
File Edit View Run Tools Help

<untitled> * x
1 # Iterative factorial program
2 num = int(input("Enter a number to calculate its factorial: "))
3 if num < 0:
4     print("Factorial is not defined for negative numbers")
5 else:
6     factorial = 1
7     for i in range(1, num + 1):
8         factorial *= i
9     print(f"The factorial of {num} is {factorial}")
10

Shell x
>>> %Run -c $EDITOR_CONTENT
Enter a number to calculate its factorial: 5
The factorial of 5 is 120
>>> |
```

Recursive Approach

Thonny - <untitled> @ 1:30

File Edit View Run Tools Help

<untitled> * x

```
1 # Recursive factorial program
2 def factorial(n):
3     if n <= 1:
4         return 1
5     return n * factorial(n - 1)
6 num = int(input("Enter a number to calculate its factorial: "))
7 if num < 0:
8     print("Factorial is not defined for negative numbers")
9 else:
10    result = factorial(num)
11    print(f"The factorial of {num} is {result}")
12
```

Shell x

```
>>> %Run -c $EDITOR_CONTENT
Enter a number to calculate its factorial: 5
The factorial of 5 is 120
>>> |
```