Exercise 86:

### Operating Mechanism of a while...else Loop

In Python, the while...else loop has the following structure:

1. **While condition is True**: The while loop checks a condition.

If the condition is **True**, it executes the block of code inside the while loop.

If the condition becomes **False**, it moves to the **else** block (if an else block exists).

1. **Else block**: The else block runs after the while loop finishes **normally** (i.e., without being interrupted by a break statement).

#### **Key points:**

* If the while loop is exited using break, the else block does **not** execute.
* The else block will only execute if the loop terminates naturally (i.e., the condition becomes False).

### Example of a while...else Loop:

# Example: Finding a number in a list

numbers = [1, 2, 3, 4, 5]

target = 6

index = 0

while index < len(numbers):

if numbers[index] == target:

print(f"Found {target} at index {index}")

break

index += 1

else:

print(f"{target} not found in the list.")

# Output:

# 6 not found in the list.

### Explanation of the Example:

* **While loop condition**: index < len(numbers) checks if there are still items left in the numbers list to check.
* **If block**: Inside the while loop, the if statement checks whether the current element (numbers[index]) matches the target (6 in this case).
  + If a match is found, the break statement stops the loop, preventing the else block from executing.
* **Else block**: If the loop finishes without finding the target (i.e., the loop condition becomes False), the else block is executed and prints a message saying that the target was not found.

In this example, since 6 is not in the list, the else block executes and prints "6 not found in the list."