**Level 4: The Looping System**

**While Loops**

**MCQs:**

1. What will happen in this loop?

x = 5

while x > 0:

print(x)

a) Infinite loop ✅  
b) Prints 5 once  
c) Error  
d) Nothing

1. What will be the output?

x = 0

while x < 3:

print(x)

x += 1

a) 0 1 2 ✅  
b) 1 2 3  
c) 2 1 0  
d) Nothing

**Fill in the Blanks:**

1. A while loop runs as long as the \_\_\_\_\_\_\_\_\_\_ is True. (condition)
2. The while loop may cause an \_\_\_\_\_\_\_\_\_\_ loop if the condition never becomes False. (infinite)

**True/False:**

1. while loops always execute at least once. (False ❌)

**For Loops**

**MCQs:**

1. What does range(5) generate?  
   a) [1, 2, 3, 4, 5]  
   b) [0, 1, 2, 3, 4] ✅  
   c) [1, 2, 3, 4]  
   d) None
2. Which loop is used when we know the number of iterations?  
   a) while  
   b) for ✅  
   c) do-while  
   d) None

**Fill in the Blanks:**

1. The for loop in Python iterates over a \_\_\_\_\_\_\_\_\_\_. (sequence)
2. range() generates a sequence of numbers starting from \_\_\_\_\_\_\_\_\_\_. (0)

**True/False:**

1. The for loop can iterate over strings. (True ✅)

**Loop Control**

**MCQs:**

1. What does the break statement do?  
   a) Skips the next iteration  
   b) Stops the loop completely ✅  
   c) Restarts the loop  
   d) Does nothing
2. What does the continue statement do?  
   a) Stops the loop  
   b) Skips the current iteration ✅  
   c) Restarts the loop  
   d) None

**Fill in the Blanks:**

1. The break statement is used to \_\_\_\_\_\_\_\_\_\_ a loop prematurely. (exit)
2. The continue statement skips the \_\_\_\_\_\_\_\_\_\_ of a loop. (current iteration)

**True/False:**

1. The continue statement stops the loop. (False ❌)