# Loops

A loop statement allows us to execute a statement or group of statements multiple times
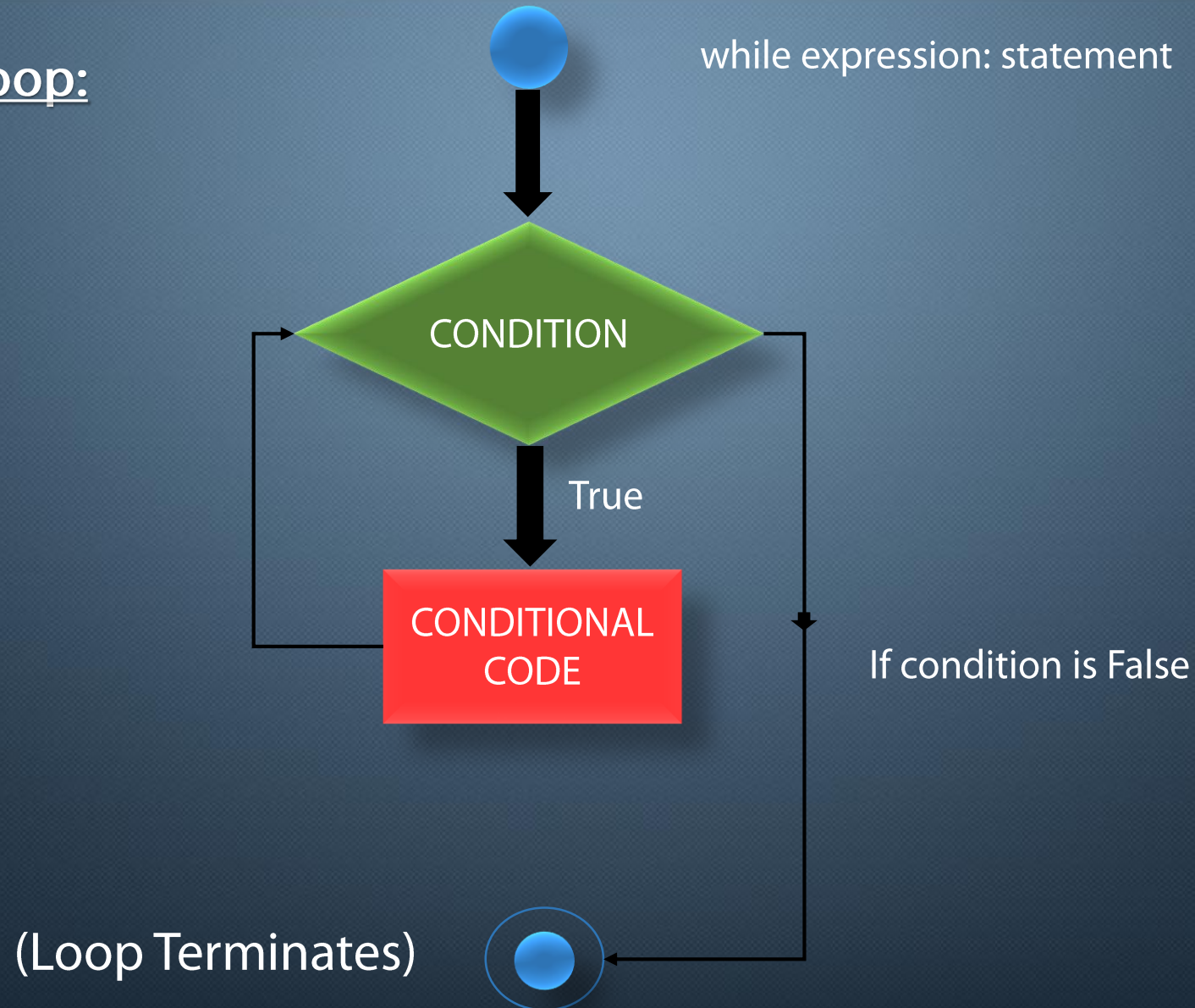
**while loop**

**for loop**

# while loop

❑ It repeatedly executes a block of (normally indented) statements as long as a test at the top keeps evaluating to a true value.

❑ Control keeps looping back to the start of the statement until the test becomes false.

## General loop format

```
while test :           #Loop Test
        statements     # Loop body
else :                 #Optional else
        statements     # Run if didn't exit loop with break
```

# while loop

**Flow of while loop:**

while expression: statement



CONDITION

True

CONDITIONAL CODE

If condition is False

(Loop Terminates)

# Statements in while loop

## Pass:

- The pass statement is a no-operation placeholder that is used when the syntax requires a statement.

## Continue:

- The continue statement causes an immediate jump to the top of a loop.

# Statements in while loop

## Break:

- The break statement causes an immediate exit from a loop

## Loop else:

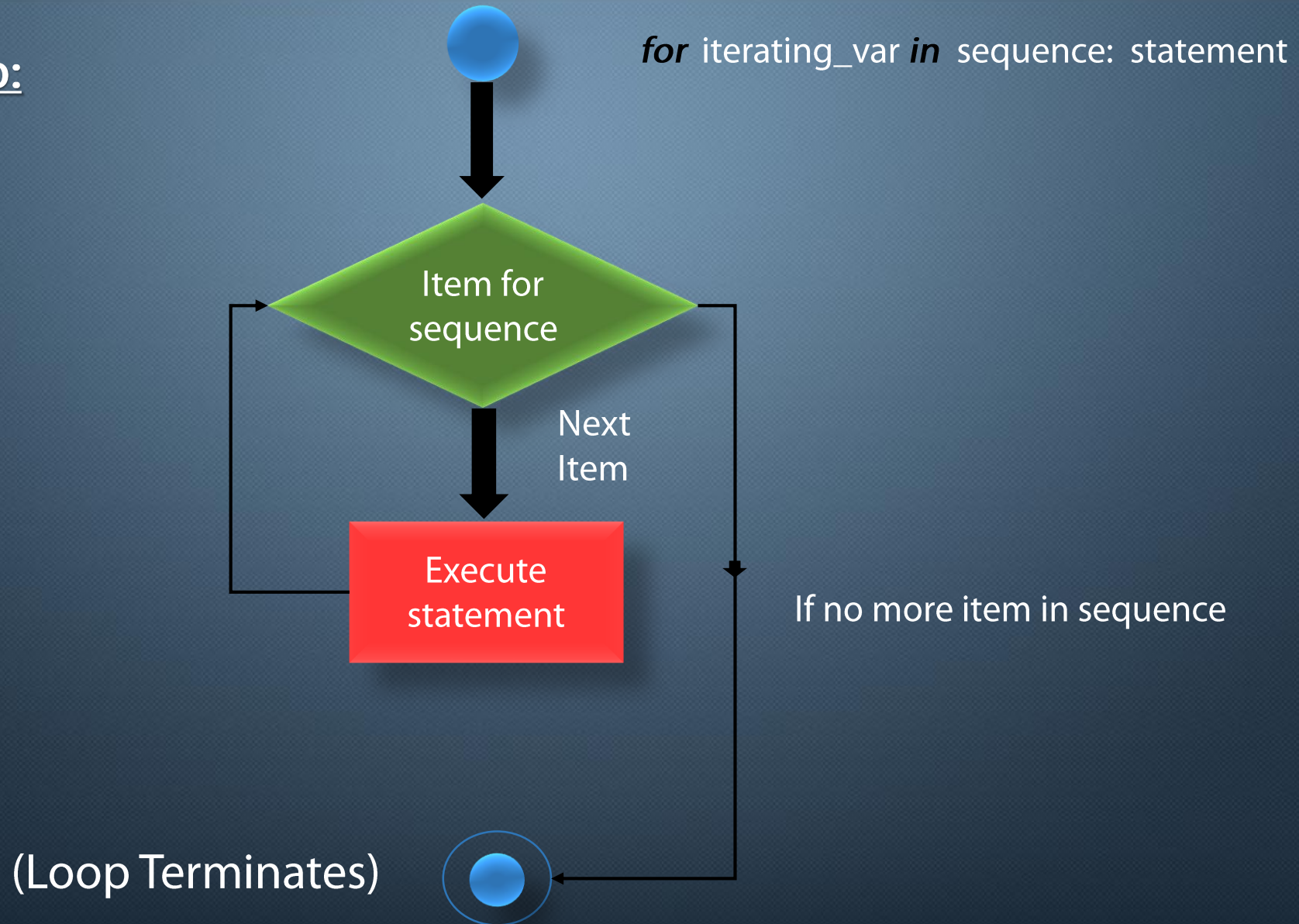- Runs if and only if the loop is exited normally.

# Hands-On

# for loop

❑ The for statement works on strings, lists, tuples and other built-in iterables, as well as new user-defined object.

❑ It can step through the items in any ordered Sequence or other iterable object.

## General loop format

```
for target in object :            #Assign object items to target
        statements                #Repeated loop body; use target
else :                            #optional else part
        statements                #if we didn't hit a 'break'
```

# for loop

**Flow of for loop:**

*for* iterating_var *in* sequence: statement

Item for sequence

Next Item

Execute statement

If no more item in sequence

(Loop Terminates)