GOVERNMENT OF WESTERN AUSTRALIA

TAFE WA South Metropolitan

For Loops in Python

(Session 7)

Review

- while Loop - executes a block of statements repeatedly until a given condition is True. For example:

```
# This Python program illustrates while loop
# The body of the loop prints the current value of the
# counter and then increments the value of counter
count = 0
while (count < 4):
    print("Count is:", count)
    count = count + 1
```

```
Count is: 0
Count is: 1
Count is: 2
Count is: 3
```

Overview

- for Loop Introduction
- for Loop Syntax
- for Loop Flowchart
- The range() Function
- The break statement
- The continue statement
- for-else loop

for Loop Introduction

- Used for iterating over a sequence
- Does not require an indexing variable
- You can use **range**() function to specify the starting value, ending value and increment
- You can use the **break** statement to stop loop
- You can use the **continue** statement to stop the current iteration of the loop

For Loop Syntax

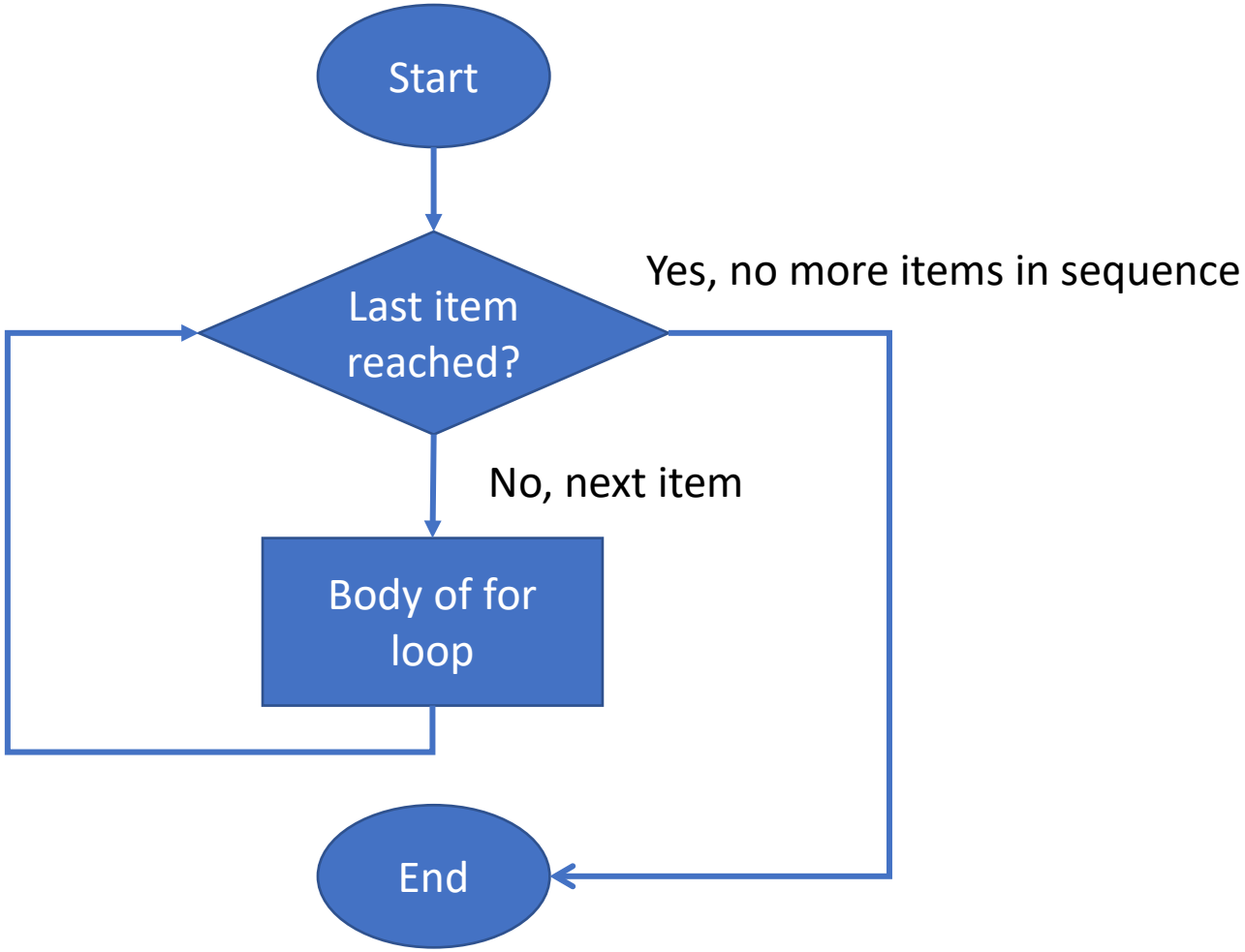
- Syntax of for loop:

for *iterating_var* in sequence**:**

    body of for loop (block of statements)

- for loop iterates over the items of any sequence
- The first item is assigned to the iterating variable
- The statement block is executed until the entire sequence is exhausted

for Loop Flowchart



for Loop Example

```
# for loop in python
# looping through a string
# prints the letters in the world "Monty Python“
for i in "Monty Python":
    print("Letter", i)

print("\nFor loop ended")
```

```
Letter M
Letter o
Letter n
Letter t
Letter y
Letter
Letter P
Letter y
Letter t
Letter h
Letter o
Letter n

For loop ended
```

The range() Function

- returns a sequence of numbers
- Syntax:

  range(*start, stop, step*)
- *start* – optional, default is 0
- *stop* – required
- *step* – optional, default is 1
- For example,
- range(3) – creates a sequence of numbers from 0 to 2
- range(2, 5) – creates a sequence of numbers from 2 to 4
- range(1, 10, 3) – creates a sequence of numbers from 1 to 7, increment is 3

```
0
1
2

2
3
4

1
4
7
```

Activity 1: create a sequence of numbers from 10 to 101, increment by 10, and print each item in the sequence

```
# for loop in python
# using range() function
# increment by 10
num_sequence = range(10, 101, 10)
for i in num_sequence:
    print(i)
```

```
10
20
30
40
50
60
70
80
90
100
```

The break statement

- Stops the loop
- Example:

```
# looping through the string variable
# prints the letter until character is " "
string = "Monty Python"
for i in string:
    if i != " ":
        print(i)
    else:
        break
```

```
M
o
n
t
y
```

The continue statement

- Stops all the statement in the current iteration. Example:

```
# looping through a string
# continue when later is digit
string = "2python_variable"
print("A variable name cannot start with a number: ", string)

for letter in string:
    if letter == "2":
        continue
    else:
        print (letter, end= "")
print(" -- correct")
```

```
A variable name cannot start with a number:  2python_variable
python_variable -- correct
```

else in for Loop

- Similar to while loops, for loops can also have an else clause
- The else keyword will be executed when all iterations are completed
- Example:

```
max_attempts = 2
for x in range(0, max_attempts):
    print ("Submission no {} ".format(x+1))
else:
    print ("\nAfter {} assessment attempts".format(x+1))
    print ("\nNot satisfactory.")
    print("Students will be required to repeat the unit.")
```

```
Submission no 1
Submission no 2

After 2 assessment attempts...

Not satisfactory.
Students will be required to repeat the unit.
```

Introduction to Data Structures

- **List**
  - Used to store multiple items
  - Lists are ordered, mutable (changeable) and allow duplicate objects
  - Example:

```
fruitlist = ['lemon', 'banana', 'kiwi', 'strawberry']
print(fruitlist)
```

```
['lemon', 'banana', 'kiwi', 'strawberry']
```

- **Tuples**
  - Used to store collection of data (similar to lists)
  - Tuples are ordered and allow duplicates, but immutable
  - Example:

```
num_sequence = (2, 3, 1, 4, 9, 6, 2, 1)
print(num_sequence)
```

```
(2, 3, 1, 4, 9, 6, 2, 1)
```

Introduction to Data Structures (cont.)

- **Sets**
  - Used to store multiple items
  - Sets are unordered, immutable (unchangeable), unindexed and do not allow duplicates
  - Example:

```
powerball = {12, 2, 18, 27, 24, 28, 6}
print("Draw 2999: ", powerball)
```

```
Draw 2999:  {2, 18, 6, 24, 27, 12, 28}
```

- **Dictionaries**
  - Used to store data values in *key*:*value* pairs
  - Dictionaries are ordered (Python 3.7 and higher), do not allow duplicates, and changeable
  - Example:

```
capitals = {"France":"Paris", "UK":" London", "USA":"Washington DC",  "Australia":"Canberra"}
print(capitals["Australia"])
```

```
Canberra
```

Activity 2: Print each fruit in a fruit list

```
# this programs uses a for loop
# to iterate over the fruitlist list
fruitlist = ['lemon', 'banana', 'kiwi', 'strawberry']
print("SMOOTHIE\n")
print ("Cut the following fruits into small pieces ", end="")
print("and place them in a blender.\n")
for fruit in fruitlist:
    print(fruit)
```

```
SMOOTHIE

Cut the following fruits into small pieces and place them in a blender.

lemon
banana
kiwi
strawberry
```

Questions?