Chapter 29

FOR LOOPS

Python uses different types of **LOOPS** to perform different actions. ***FOR* LOOPS** repeat code a specified number of times.

LOOP
Code used to repeat a sequence of commands

FORMATTING *FOR* LOOPS

The *for* loop follows a very specific format; even the number of spaces you use on certain lines is important.

Steps for Creating a *For* Loop:

STEP 1: Enter the key word "for." *For* loops always start with "for" to show that the following code is a *for* loop.

STEP 2: Name the **COUNTER VARIABLE**. This variable's value increases each time the loop repeats. For example, if you are using a *for* loop that loops 10 times, starting with 0 and increasing by 1 each time the loop starts over, the counter variable will represent 0 in the first run, then 1 in

the next run, then 2, then 3, and so on, increasing by 1 each time the loop starts over.

You can name the counter variable anything you want. Many programmers name their counter variable "i" out of tradition.

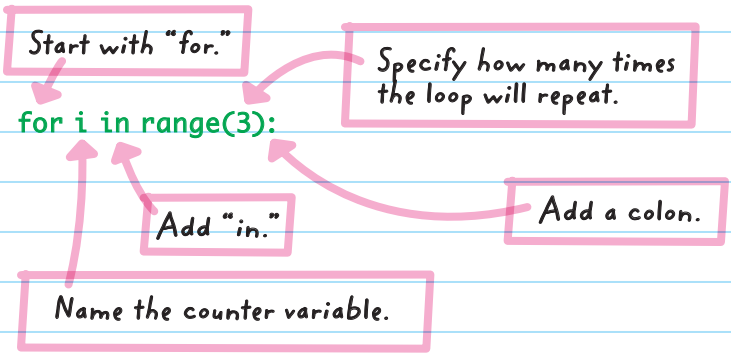
STEP 3: Add the key word "in" to show that you are about to specify how many times the loop should repeat.

STEP 4: Set the number of times the loop should repeat. Use the **range()** function to do this. The range function will count from zero up to, but not including, the number listed in the function's parentheses.

Range() defines where a counter starts, ends, and what number it should count by. The parameters include (start, stop, step). The **start** and **stop** parameters name the start and stop numbers for the counter. The **step** parameter tells how much the counter should count by. For example, range(4, 13, 4) will count from 4 to 12 (the number before the "stop" number) in increments of 4; the numbers counted in this range are: 4, 8, 12.

STEP 5: End the first line of the *for* loop with a colon (:).

Here's what a *for* loop, with a counter variable named *i*, counting by 1 from 0 up to 3, looks like:



Below the first line of the *for* loop, add the code to be repeated.

Indent the code to be repeated by pressing the **TAB** key on the keyboard once or by using the space bar to add FOUR SPACES. In Python, the indent indicates that the code on that line is part of the code above it. You can add a print

function to the loop by starting a new line and indenting it in the same way.

FOR EXAMPLE, you could add a print function that displays "Hip hip hooray!"

```
for i in range(3):
    print("Hip hip hooray!")
```

Insert the code you want repeated.

Indent one tab (four spaces).

This will print:

```
Hip hip hooray!
Hip hip hooray!
Hip hip hooray!
```

You can add code that runs after the loop has finished running.

To do that, start a new line that's not indented, like this:

```
for i in range(3):
    print("Hip hip hooray!")
print("Congratulations")
```

In the loop

Outside the loop

This will print:

```
Hip hip hooray!
Hip hip hooray!
Hip hip hooray!
Congratulations
```

Every time you create a *for* loop, you also create a new variable that counts the number of times the loop is repeated. That means the variable can be used within the repeated code.

In the previous example, the variable "i" was used in the *for* loop. Printing the value of "i" before "Hip hip hooray!" shows how the value increases each time the loop is run:

```
for i in range(3):
    print(i, "Hip hip hooray!")
```

REMINDER: The range( ) function starts counting at 0 and stops counting before the number set in the parentheses.

This will print:

```
0 Hip hip hooray!
1 Hip hip hooray!
2 Hip hip hooray!
```

In the display output, the "i" variable value in the first run of the loop is 0 and the last run shows "i" having the value of 2.

The counter variable can be used to count in other ways.

FOR EXAMPLE, you could make a countdown. To do this, subtract the counter variable from 10, and the computer will display a countdown from 10 to 1:

```
for i in range(10):
    print(10 - i)
```

This will print:

```
10
9
8
7
6
5
4
3
2
1
```

The range() function is set to 10, but can go as high as you want it to.

You could use the other parameters in the range() function to count by 2s between 2 and 10:

```
for i in range(2, 10, 2):
    print(i)
```

Start at this number.

Count (step) by this number.

End before this number.

This will print:

```
2
4
6
8
```

To print every item in a list (like "fruits") use this code:

```
fruits = ['pears', 'oranges', 'mangos', 'cherries', 'bananas', 'apples']

for i in range(6):
    print(fruits[i])
```

item in list

list name

This would print:

```
pears
oranges
mangos
cherries
bananas
apples
```

There's an easier way to loop through an entire list without having to know the exact length of the list.

Instead of using the **range()** function, use the list itself, and the loop will run the same amount of times as the number of items in the list. Instead of being a number, the counter variable value will be set to each item of the list as it steps through the entire list.

To print every fruit in the "fruits" list, replace range() with the list name:

```
fruits = ['pears', 'oranges', 'mangos', 'cherries', 'bananas',
'apples']

for i in fruits:
    print(i)
```

Use the list name instead of range().

The variable value of i is the list item, not a number.

CHECK YOUR KNOWLEDGE

1. Explain what the range() part of a *for* loop does.

2. Which part of a *for* loop's code should be indented?

3. What are two ways to print each item in the following list?

```
Seasons = ["Winter", "Spring", "Summer", "Fall"]
```

Turn the page for question #4

4. Write the output for each of the programs below:

PROGRAM NAME	PROGRAM	RESULT
A	`for i in range(4):` `    print(i)`	
B	`for i in range(10):` `    print(i * 5)`	
C	`for i in range(25, 101, 25):` `    print(i)`	

D	`colors = ["red", "blue",`
	`"yellow"]`
	`for i in colors:`
	`    print(i)`
	`print("Those are the primary`
	`colors")`	

E	`multiplier = 4`
	`for i in range(11):`
	`    print(i * multiplier)`	

F	`for i in range(7):` `    print(i)`	

CHECK YOUR ANSWERS

1. The range() function defines where the loop counter starts, ends, and what number it should count by.

2. The code that's four spaces in, or tabbed, is the part of the code that runs each time the *for* loop cycles.

3. One way:

```
for i in range(4):
    print(Seasons[i])
```

Another way:

```
for i in Seasons:
    print(i)
```

4. Code results:

A: 0
1
2
3

B: 0
5
10
15
20
25
30
35
40
45

C: 25
50
75
100

D: red
green
blue
yellow
Those are the primary colors

E: 0
4
8
12
16
20
24
28
32
36
40

F: 0
1
2
3
4
5
6