THE FOR LOOP

The for..in statement is another looping statement which **iterates** over a sequence of objects i.e. go through each item in a sequence. We will see more about sequences in detail in later chapters. What you need to know right now is that a sequence is just an ordered collection of items.

Example (save as for.py):

```
for i in range(1, 5):
    print i
else:
    print 'The for loop is over'
```

Output:

```
$ python for.py
1
2
3
4
```

```
The for loop is over
```

How It Works

In this program, we are printing a **sequence** of numbers. We generate this sequence of numbers using the built-in range function.

What we do here is supply it two numbers and range returns a sequence of numbers starting from the first number and up to the second number. For example, range(1,5) gives the sequence [1, 2, 3, 4]. By default, range takes a step count of 1. If we supply a third number to range, then that becomes the step count. For example, range(1,5,2) gives[1,3]. Remember that the range extends **up to** the second number i.e. it does **not** include the second number.

Note that range() generates a sequence of numbers, but it will generate only one number at a time, when the for loop requests for the next item. If you want to see the full sequence of numbers immediately, use list(range()). Lists are explained in the data structures chapter.

The for loop then iterates over this range - for i in range(1,5) is equivalent to for i in [1, 2, 3, 4] which is like assigning each number (or object) in the sequence to i, one at a time, and then executing the block of statements for each value of i. In this case, we just print the value in the block of statements.

Remember that the else part is optional. When included, it is always executed once after the for loop is over unless abreak statement is encountered.

Remember that the for..in loop works for any sequence. Here, we have a list of numbers generated by the built-inrange function, but in general we can use any kind of sequence of any kind of objects! We will explore this idea in detail in later chapters.

NOTE

Note for C/C++/Java/C# Programmers

The Python for loop is radically different from the C/C++ for loop. C# programmers will note that thefor loop in Python is similar to the foreach loop in C#. Java programmers will note that the same is similar to for (int i : IntArray) in Java 1.5.

In C/C, if you want to write `for (int i = 0; i < 5; i), then in Python you write just `for i in range(0,5). As you can see, the for loop is simpler, more expressive and less error prone in Python.

Source: http://www.swaroopch.com/notes/python/