For loops:

```
for i in range(0,10,1):
```

In python, for a simple loop, the **for** keyword is followed by a the words, “in range” followed by a set of parentheses containing three parts separated by commas.

```
for variable in range (init, max, increment):
```

The intent of this **for** loop is to enable stepping a variable through a series of numbers, like counting from 0 to 9. The part before the first comma (*init*) is the number at which the variable will start counting. The part between the two commas (*max*) is the number the variable will count to but not include. And the part following the final comma (*increment*) is what the variable will be counting by. If init and increment are not given, they are assumed to be 0 and 1 respectively.

In practice, **for** loops are used most often for counting out n iterations. The standard idiom for this is the following.

```
for i in range (n):
   body
```

Here we have a counter variable i whose value starts at 0. With each iteration, we test whether i has reached n or not; and if it hasn't, then we execute the **for** statement's body and then i goes to the following integer. The result is that the body is executed for each value of i from 0 up to n – 1. Note: Python uses whitespace to differentiate between the body and something outside of the **for** loop that happens afterwards. That is, every statement you want to be repeated must be indented below the **for** statement.

But you can use a **for** loop for other purposes, too. In the following example, we display the multiples of 2 up to (but not including) 12. Notice how the *increment* portion of the **for** statement has changed to 2.

```
for p in range(0,12,2):
   print(p)
```

While loops:

```
while (test):
    body
```

The **while** statement works by checking the test condition, if it is true then the body will be executed. After the body is executed it will check the test condition again and repeat until the test condition is no longer true.

```
i=10
while (i >= 0):
  print(i)
  i-=1
```

Again, Python uses whitespace to differentiate between the body and something outside of the **while** loop that happens afterwards. That is, every statement you want to be repeated must be indented below the **while** statement.