حلقه‌های While

حلقه `while` برخلاف حلقه for بجای تکرار روی یک دنباله، روی برقراری شرایط کار می‌کند، یعنی «مادامی که شرط برقرار باشد حلقه در جریان است.» ساختار این حلقه در پایتون به صورت زیر است:

```
while test_conditoin:
        code statements
    else:
        final code statements
```

In [6]:
```
x = 0
while x < 10:
    print('x is currtenly: ', x)
    print(' x is still less than 10, adding 1 to x')
    x += 1
```

```
x is currtenly:  0
 x is still less than 10, adding 1 to x
x is currtenly:  1
 x is still less than 10, adding 1 to x
x is currtenly:  2
 x is still less than 10, adding 1 to x
x is currtenly:  3
 x is still less than 10, adding 1 to x
x is currtenly:  4
 x is still less than 10, adding 1 to x
x is currtenly:  5
 x is still less than 10, adding 1 to x
x is currtenly:  6
 x is still less than 10, adding 1 to x
x is currtenly:  7
 x is still less than 10, adding 1 to x
x is currtenly:  8
 x is still less than 10, adding 1 to x
x is currtenly:  9
 x is still less than 10, adding 1 to x
```

Notice how many times the print statements occurred and how the `while` loop kept going until the True condition was met, which occurred once x==10. It's important to note that once this occurred the code stopped. Let's see how we could add an `else` statement:

In [7]:
```
x = 0

while x < 10:
    print('x is currently: ',x)
    print(' x is still less than 10, adding 1 to x')
    x+=1

else:
    print('All Done!')
```

```
x is currently:  0
 x is still less than 10, adding 1 to x
x is currently:  1
 x is still less than 10, adding 1 to x
x is currently:  2
 x is still less than 10, adding 1 to x
x is currently:  3
 x is still less than 10, adding 1 to x
x is currently:  4
 x is still less than 10, adding 1 to x
x is currently:  5
 x is still less than 10, adding 1 to x
x is currently:  6
 x is still less than 10, adding 1 to x
x is currently:  7
 x is still less than 10, adding 1 to x
x is currently:  8
 x is still less than 10, adding 1 to x
x is currently:  9
 x is still less than 10, adding 1 to x
All Done!
```

break, continue, pass

ما می‌توانیم درون حلقه‌ها شرایطی را فراهم کنیم که حلقه بشکند، ادامه پیدا کند و یا بی‌تأثیر باشد. تعاریف زیر را ببینید:

```
break: Breaks out of the current closest enclosing loop.
continue: Goes to the top of the closest enclosing loop.
pass: Does nothing at all.
```

Thinking about `break` and `continue` statements, the general format of the `while` loop looks like this:

```
while test:
    code statement
    if test:
        break
    if test:
        continue
else:
```

`break` and `continue` statements can appear anywhere inside the loop's body, but we will usually put them further nested in conjunction with an `if` statement to perform an action based on some condition.

Let's go ahead and look at some examples!

In [9]:
```
x = 0

while x < 10:
    print('x is currently: ',x)
    print(' x is still less than 10, adding 1 to x')
    x+=1
    if x==3:
        print('x==3')
    else:
        print('continuing...')
        continue
    print('dsdsdssd')
```

```
x is currently:  0
 x is still less than 10, adding 1 to x
continuing...
x is currently:  1
 x is still less than 10, adding 1 to x
continuing...
x is currently:  2
 x is still less than 10, adding 1 to x
x==3
dsdsdssd
x is currently:  3
 x is still less than 10, adding 1 to x
continuing...
x is currently:  4
 x is still less than 10, adding 1 to x
continuing...
x is currently:  5
 x is still less than 10, adding 1 to x
continuing...
x is currently:  6
 x is still less than 10, adding 1 to x
continuing...
x is currently:  7
 x is still less than 10, adding 1 to x
continuing...
x is currently:  8
 x is still less than 10, adding 1 to x
continuing...
x is currently:  9
 x is still less than 10, adding 1 to x
continuing...
```

Note how we have a printed statement when x==3, and a continue being printed out as we continue through the outer while loop. Let's put in a break once x ==3 and see if the result makes sense:

In [11]:
```
x = 0

while x < 10:
    print('x is currently: ',x)
    print(' x is still less than 10, adding 1 to x')
    x+=1
    if x==3:
        print('Breaking because x==3')
        break
    else:
        print('continuing...')
        continue

else:
    print('All Done.')
```

```
x is currently:  0
 x is still less than 10, adding 1 to x
continuing...
x is currently:  1
 x is still less than 10, adding 1 to x
continuing...
x is currently:  2
 x is still less than 10, adding 1 to x
Breaking because x==3
```

Note how the other `else` statement wasn't reached and continuing was never printed!

After these brief but simple examples, you should feel comfortable using `while` statements in your code.

A word of caution however! It is possible to create an infinitely running loop with `while` statements. For example:

In [ ]:
```
# DO NOT RUN THIS CODE!!!!
while True:
    print("I'm stuck in an infinite loop!")
```

A quick note: If you *did* run the above cell, click on the Kernel menu above to restart the kernel!