Breaking Out of Loops

Breaking Out of Loops

Sometimes you want to exit a loop before it's finished, or skip the remainder of a loop and move to the next iteration. To do this you can use the `break` and `continue` statements respectively.

`break`

As a first example, consider:

[1]:
```
for i in range(10):
    print(i)

    if i == 5:
        break
```

```
0
1
2
3
4
5
```

where you can see that the loop terminated before it was finished iterating through `range(10)`. The `break` may be inside the `if` statement, but it's the loop that it affects.

The `break` statement exits the first loop that it's nested in. For example, if we had multiple nested loops:

[5]:
```
for i in range(3):
    print('Loop1', i)
    for j in range(3):
        print('    Loop2', j)

        if j == 1:
            break
```

```
Loop1 0
    Loop2 0
    Loop2 1
Loop1 1
```

```
    Loop2 0
    Loop2 1
Loop1 2
    Loop2 0
    Loop2 1
```

We can see that the outer loop (Loop1) iterated through all of `range(3)`, while Loop2 terminates before it can reach the last iteration.

`continue`

If you want to end the current loop iteration, but you don't want to break out of the loop, you can use the `continue` statement.

[9]:
```
for i in range(10):
    if i == 5:
        continue
    print(i)
```

```
0
1
2
3
4
6
7
8
9
```

As you can see in the example above, `5` is not printed.