Week 4: Iteration - for loops

- range() see separate document

- **for** loops
- Nested iteration

for loops

- A **for loop**
  - Also known as a **count controlled** loop.
  - Repeats part of a program for a stated number of times.
  - Repeats its loop body for each element of the sequence in order.
  - The loop ends when it reaches the number of times it was set to loop.

Counting with For loops

In Python, for loops are constructed like this:

```
for [iterating variable] in [sequence]:
    [do something]
```

- The **for** and **in** are key words that are inbuilt commands.
- The iterating variable will **hold the values of the generated sequence** which is stepped through. E.g.,

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

Examples of **for** Loops

Table 2 for Loop Examples

Loop	Values of i	Comment
`for i in range(6) :`	0, 1, 2, 3, 4, 5	Note that the loop executes 6 times.
`for i in range(10, 16) :`	10, 11, 12, 13, 14 15	The ending value is never included in the sequence.
`for i in range(0, 9, 2) :`	0, 2, 4, 6, 8	The third argument is the step value.
`for i in range(5, 0, -1) :`	5, 4, 3, 2, 1	Use a negative step value to count down.

Nested iteration

```
for i in range(3):
    print(i)
```

- We know the value of `i` will be 0, then 1, then 2. The print will be performed once for each pass.
- A **nested iteration** is a loop within a loop. E.g.,

```
for i in range(3):
    for j in range(2):
        print(i, j)
```

- We could call these the **outer iteration** and the **inner iteration**.

Nested iteration

```
for i in range(3):          # 0,1,2
    for j in range(2):      # 0,1
        print(i, j)
```

- The program first encounters the outer loop (for i), executing its first iteration (0). The first iteration triggers the **inner loop** (for j) **which then runs to completion.**
- Program then returns back to **outer loop** for next iteration.
- For each value of `i`, all values of `j` will occur:

```
0 0
0 1
1 0
1 1
2 0
2 1
```

Week 4: Iteration - for loops

- **for** loops
- Nested iteration