Python: Decisions and Loops*

Arash Pourhabib†

1 Branching

Branching: if a condition is met we do task A, otherwise we do task B. Example:

$$f(n) = \begin{cases} \frac{n}{2} & \text{if } n \text{ is even} \\ -\frac{n-1}{2} & \text{if } n \text{ is odd} \end{cases}$$

In Python, we use **if-else**

```
if condition:
    <block of statements, executed if condition is True>
else:
    <block of statements, executed if condition is False>
```

The block of statements must be indented (use Tab key once, or hit Space four times).

The **else** part of an **if** test can be skipped, if not needed.

*References: (1) Langtangen, Hans Petter. A primer on scientific programming with Python, Fourth Edition. Springer, 2014. (2) https://docs.python.org/2/reference/

†School of Industrial Engineering and Management, Oklahoma State University

2 Branching

Kilograms to pounds conversion

```
K = 100
if K < 0:
    print('Weights cannot be negative!')
else:
    P = 2.204 * K
    print(P)
```

- Use **elif**, short for "esle if", for several mutually exclusive if tests

```
if condition:
    <block of statements>
elif condition2:
    <block of statements>
elif condition3:
    <block of statements>
else:
    <block of statements>
```

3 Boolean operators

Other examples for comparison in Python:

```
K == 10     # K equals to 10
K != 10     # K does not equal to 10
K < 10     # K is less than 10
K <= 10     # K is less than or equal to 10
K > 10     # K is gretaer than 10
K >= 10     # K is gretaer than or equal to 10
```

4 Loops

- Loops are used to efficiently express repetition
- Tow variants of loops in Python: **while** and **for**

While loop repeats a set of statements as long as a condition is true. Example:

```
K = 0     #initial value
dK = 10     #increment
while K < 200:     #loop heading with condition
    P = 2.204 * K     # 1st statement
    print('{:.2f} kg = {:.2f} lb'.format(K,P)) #2nd statement
    K = K + dK     #3rd statement: increment counter
print('outside of while loop')
```

Like **if**, indentation defines a block. Unlike VBA or MATLAB, there is no "end" command.

5 For Loop

- **for** is used to repeat tasks for a "known" or "fixed" number of times
- You can iterate over a list or use Python **range**

Example 1: iterating over a list

```
kilos= [100, 120, 130, 150, 200]
for K in kilos:
    P = 2.204 * K
    print('{:.2f} kg = {:.2f} lb'.format(K,P))
```

Example 2: iterating using **range**

```
n = 10
for K in range(n):
    P = 2.204 * K
    print('{:.2f} kg = {:.2f} lb'.format(K,P))
```

range(n) generates integer 0, 1, 2, ..., n-1.

range(start,stop,step) generates integer `start`, `start+step`, `start+2*step`, up to, but not including, `stop`.

6 Zip function

zip is used to join two lists for iterations

```
kilos = [60., 66., 73., 81]
pounds = [132.2, 145.5, 160.9, 178.6]

for k, p in zip(kilos, pounds):
    print('{:.1f} kilograms is equal to {:.1f} pounds'.format(k, p))
```