Operators and strings

Python can be used as a calculator. Standard operations include +,-,*,/,**,//,%. Let's see what these do:

In [1]:

```
3+7
5+8
```

Out[1]:

```
13
```

In [5]:

```
print("Addition: with '+': \t\t3 + 11 = ", 3 + 11)
print("Subtraction with '-': \t\t7 - 11 = ", 7 - 11)
print("Multiplication with '*': \t78.2 * 99.3 = ", 78.2 * 99.3)
print("Division with '/': \t\t38 / 14 = ", 38/14)
print("Exponentiation with **: \t2**10 = ", 2**10)
```

```
Addition: with '+':             3 + 11 =  14
Subtraction with '-':           7 - 11 =  -4
Multiplication with '*':        78.2 * 99.3 =  7765.26
Division with '/':              38 / 14 =  2.7142857142857144
Exponentiation with **:         2**10 =  1024
```

Notice a few things here. We used the "print"-function, which prints all kinds of things and tries to make the output as nice as possible. The first thing printed here are `strings`, which are characters enclosed in quotes. Strings can also be manipulated by some of the operations above:

In [3]:

```
print("first part" + "second part")
print(3*"three times")
print("number is " + str(3))    # concatenation only works with strings
```

```
first partsecond part
three timesthree timesthree times
number is 3
```

The `\t` are tab characters, which help align output nicely. To concatenate numbers to a string, you need to use the `str()` function on it first.

We missed two of the operators mentioned above. What do they do?

In [1]:

```
print("??: \t67 // 11 = ", 67 // 11)
print("??: \t67  % 11 = ", 67  % 11)
```

```
??:     67 // 11 =  6
??:     67  % 11 =  1
```

A convenient way to add or subtract things is using the following abbreviated operators:

In [2]:

```
a = 17

a *= 7 # a = a  + 7
print(a)

a -= 7
print(a)
```

```
119
112
```