Python Programming Tasks **Data Mining Practical Course Summer 2018**
Hints and Solutions

PART I

This task is the "very beginners" task for all who are using Python the first time, that is, and reduces to use Python as a very sophisticated calculator.

However, be aware of differences between Python 2.7.x versions and Python 3.x, especially for integer division, which you may not even notice in a bigger program:

Python 2.7.12	Python 3.4.1
`3/2 = 1` integer division results in integer)	`3/2 = 1.5` (integer division results in float)
`3/2.0 = 1.5`	`3/2.0 = 1.5`

Clearly, more functionalities, e.g. the `print` command, are affected when porting code between the versions. It is beyond of the scope of our tutorial, but who wants to proceed using or writing Python code should pay some attention on this issue. An overview is provided e.g. here:

`http://python-future.org/compatible_idioms.html`

If you want to know the actual installed Python version, simply type:

```
$ python -V
Python 2.7.12
```

The remainder of this task is basically a refresh of common mathematical rules (e.g. * binds stronger than +) and on boolean expressions.

Tasks 2-4 deals with (multiple) variable assigment, and processing of strings and lists.

```
>>> x=4
>>> print x
4
>>> y = x+1
>>> print y
5
>>> x=2
>>> print x, y
2 5
```

When executing assignments, python evaluates the right hand side first and then assigns those values to the variables specified in the left hand side. Of course, the assigment "=" is asymmetric:

```
4=x
File "<stdin>", line 1
SyntaxError: can't assign to literal
```

Python allows multiple variable assignment

```
>>> x,y = 2, 6
>>> print x, y
2 6
>>> x, y = y, x+2
>>> print x, y
6 4
```

For the list task it is basically important to address specific list members, e.g. the second entry in a list , e.g `c[1]`. Note, that the first element is addressed as `c[0]` as is standard in most high-level programming languages, but may differ for e.g. Matlab software. Also, be aware that

`c[0:2]` returns `[1, 2]`

Using this command "slices" your list (or your matrix) and goes up to, but does not include, the second index (here `2`)

`c[-1]` is negatively indexing your list, i.e. starting from the last element, and returns `8`

The `range` command is important when using e.g. a `for` loop:

`range(2, 12, 3)` returns `[2, 5, 8, 11]`, i.e. integers between 2 and 12 in steps of 3

PART II

- `A=numpy.array([[1],[2],[3]])`
- `B[:,1:3]`
- `numpy.ones((3,1))*5`
- `numpy.zeros(2,3)`

PART III

Please find all functions collected in the Python script 'DAMI1-Part3.py'