INF5830 2013: Python and Numpy – some hints

random

Working with statistics, we may at some point want to pick elements or sets of elements at random. The Python library contains a module for that, called random, see
http://docs.python.org/2/library/random.html
Test out some of the functionality, e.g. the following commands (including the repetitions)

```
>>> a = range(100)
>>> a
>>> random.choice(a)
>>> import random
>>> random.choice(a)
>>> random.choice(a)
>>> random.sample(a,10)
>>> random.sample(a,10)
>>> random.shuffle(a)
>>> a
```

Make sure you understand the commands.

Importing modules and name spaces

We do not have access to random.choice() before we have imported the module random. Another possibility is to import the function random.choice itself, or all functions from random, say

```
>>> a = range(100)
>>> from random import *
>>> choice(a)
>>> sample(a,20)
>>> shuffle(a)
>>> a
```

This is convenient because it saves us from typing. But there is a danger. If another module uses the same names for classes or functions, we get a name conflict and we cannot access both functions under the same name. Each module has its own name space and if we use the longer name including the module name, we are less likely to experience conflicts.

A third possibility, to save us some writing is to import a module under a shorter alias name, try e.g.,

```
>>> a = range(100)
>>> import random as rd
>>> rd.choice(a)
>>> random.choice(a)
```

You may read more in the Python documentation or in ch. 10 of How to think like a computer scientist.

NLTK

is another library. We can neither call ChartParser() nor nltk.ChartParser() before we have imported nltk. After we have imported nltk we can use nltk.ChartParser() but not ChartParser(). If we add

```
>>> from nltk import ChartParser
>>> ChartParser()
```

should work.

NumPy

is a tool for scientific computing with Python. It adds both functionality and speed. The basic additional brick is the N-dimensional array data type. We will mainly consider one-dimensional arrays. A one-dimensional array is similar to a list but:

- All elements must be of the same type
- It has a fixed length in the sense that we do not append or remove elements from it
- It has additional methods and functionality

We may make a new array in many ways, e.g.

```
>>> import numpy as np
>>> a = range(100)
>>> b = np.array(a)
>>> c = np.arange(100)
>>> b
>>> c
>>> type(a)
>>> type(b)
>>> type(c)
>>> z=np.zeros(200)
>>> z
```

Let us inspect some of the new functionality

```
>>> d = b+c
>>> e = b*c
>>> d
>>> e
>>> b+3
>>> b*3
>>> g=b/3
>>> g
>>> f = b*1.0
>>> h= f/3
>>> h
>>> h==g
>>> a+b
>>> k = range(800,900)
>>> a+k
```

Observe how numpy does type cohesion in a+b and transforms a to an array. See how this differs from the list operation in a+k.

numpy.random

Numpy has its own random module. One should beware that this module is different from the Python module random, and uses some of the same function names with a different interpretation. It is important to

- Know which name spaces you are using
- Consult the documentation for the functions before you use them.

Some tools for statistics in NumPy

The NumPy array has some built-in methods useful for statistics, e.g. consider the following

```
>>> b.mean()
>>> b.var()
>>> b.std()
```

One should beware how the variance and standard deviation is calculated, though. Consider the observation {1,2,3}. Calculate its mean, sample variance and sample standard deviation by hand. Then see what the software yields:

```
>>> c = np.array([1,2,3])
>>> c.mean()
>>> c.var()
>>> c.std()
>>> c.var(ddof=1)
```