2. Python Module NumPy

1 Klassen und Methoden

classes: Klassen definieren Datentypen in Python, sie vereinen Eigenschaften von Objekten und liefern Methoden für diesen Datentyp/diese Klasse. Sie sind die Templates für Objekte
Keyword: **class**

```
class myClass:
    def methodFromClass(instancePtr, args, kwargs):
        <Statements innerhalb der Funktion>
        return <zurueckgegebenes Argument>
```

objects: Objekte sind konstruiert aus Klassen. Das Objekt hat Zugriff auf Methoden der Klasse und Klassenattribute

```
myObject = myClass()
```

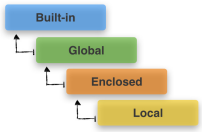
functions: kleine Codebausteine, die sich nach einmaliger Definition überall im Code aufrufen lassen und aus anderen Modulen importiert werden können
Keyword: **def**

```
def functionName(arguments, keyword arguments):
    <Statements innerhalb der Funktion>
    return <zurueckgegebenes Argument>
```

2 Namespace & Scope

Namespace: Bereich in einem Pythonskript, in dem die Namen initialisierter Objekte leben. Python sucht diese Namen entlang der folgenden Hierarchie: LERP

- **Local:** z.B. in einer Funktion
- **Enclosed:** z.B. im Falle einer Funktion innerhalb einer Funktion
- **Global:** höchstes Level des ausführenden Skripts
- **Built-in:** von Python reservierte Wörter



3 Module & Packages

Module:

- .py - Datei mit Definitionen und Statements
- hat seinen eigenen, abgeschlossenen Namespace: Namen können nicht doppelt vergeben werden (der zuletztdefinierte existiert).
- kann in andere Pythonskripte importiert werden

Import Statements:

```
import numpy
```

erlaubt den Zugriff auf alle Klassen und Funktionen in Python mit dem Prefix **numpy.**

```
import numpy as np
```

ändert den benötigten prefix auf **np.**.
Ist im Falle von Numpy die gängige Konvention.

```
from numpy import array
```

importiert nur die Klasse Array mit Name **array**

```
from numpy import array as ar
```

importiert nur die Klasse Array und ändert ihren Zugriffsnamen auf **ar**

```
from numpy import *
```

importiert die kompletten Namen von Numpy, die ohne prefix verwendet werden können. Sollte generell nicht verwendet werden.

4 NumPy

- **NumPy:** Numerical Python
- designed um Python mit C zu kombinieren, um Datenverarbeitung mit hoher Geschwindigkeit zu erreichen
- Grundlage für viele weitere Bibliotheken, die im Bereich Datenverarbeitung verwendet werden

4.1 Grundlegende Befehle

- Erzeugen von leeren Arrays: np.empty, np.ones, np.zeros, np.arange, np.random.rand
- Verändern der Dimension des Arrays: Vektor mit 9 Einträgen: np.reshape(3, 3)
- Array Slicing: Auswählen verschiedener Subarrays durch Indexauswahl

```
>>> a[0,3:5]
array([3,4])

>>> a[4:,4:]
array([[44, 45],
       [54, 55]])

>>> a[:,2]
array([2,12,22,32,42,52])

>>> a[2::2,::2]
array([[20,22,24]
       [40,42,44]])
```