Python Practise Session

Muhammad Shiraz Ahmad, Department of Physics, LUMS, Lahore, Pakistan. 27 Feb 2020

NumPy

NumPy - Array Creation Routines

In [10]:
```
%config IPCompleter.greedy=True

import numpy as np
print(np.zeros(10))
print(np.ones(10))
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [17]:
```
import random
x = np.round(np.random.rand(10,10)*10,1)
x
```

Out[17]:
```
array([[ 1.2,  2.8,  4.3,  1.8,  8.3,  7.3,  8.9,  4. ,  7.3,  9.9],
       [ 2. ,  7.9,  3.8,  2.6,  2.6,  6.4,  4.6,  4.8,  7.4,  6.2],
       [ 2.9,  2.6,  4.1,  7.3,  8.9,  3.7,  4.9,  8.3,  3. ,  8. ],
       [ 4. ,  9.2,  5.8,  8.6,  5.4,  1.2,  2.5,  9.7,  6.1,  0.8],
       [ 7.5,  0.4,  6.9,  0.2,  3.3,  6.1,  8.8,  6.2,  7.1,  7. ],
       [ 4.9,  5.1,  8.1,  2.7,  6.9,  3.1,  1.3,  3.3,  5.3,  0.9],
       [ 7.6,  3.7,  6.7,  3.7,  8.2,  9.5,  1.1,  8.3,  3.8, 10. ],
       [ 6.1,  5.8,  3.8,  0.2,  0.1,  6.7,  1.7,  8. ,  4.6,  8.2],
       [ 1.5,  2.5,  4.4,  5.8,  8.7,  8.2,  5.9,  5.6,  2.5,  6.1],
       [ 2.1,  2.8,  0.5,  2.5,  1.6,  7.8,  2.4,  6.1,  8.9,  9. ]])
```

NumPy - Indexing

In [20]:
```
x[:,0]
```

Out[20]:
```
array([1.2, 2. , 2.9, 4. , 7.5, 4.9, 7.6, 6.1, 1.5, 2.1])
```

In [21]:
```
x[:,1]
```

Out[21]:
```
array([2.8, 7.9, 2.6, 9.2, 0.4, 5.1, 3.7, 5.8, 2.5, 2.8])
```

Conditions

In [22]:
```
# Syntax x[Conditions]
x[x<1]
```

Out[22]:
```
array([0.8, 0.4, 0.2, 0.9, 0.2, 0.1, 0.5])
```

In [23]:
```
x[x>1]
```
Out[23]:
```
array([ 1.2,  2.8,  4.3,  1.8,  8.3,  7.3,  8.9,  4. ,  7.3,  9.9,  2. ,
        7.9,  3.8,  2.6,  2.6,  6.4,  4.6,  4.8,  7.4,  6.2,  2.9,  2.6,
        4.1,  7.3,  8.9,  3.7,  4.9,  8.3,  3. ,  8. ,  4. ,  9.2,  5.8,
        8.6,  5.4,  1.2,  2.5,  9.7,  6.1,  7.5,  6.9,  3.3,  6.1,  8.8,
        6.2,  7.1,  7. ,  4.9,  5.1,  8.1,  2.7,  6.9,  3.1,  1.3,  3.3,
        5.3,  7.6,  3.7,  6.7,  3.7,  8.2,  9.5,  1.1,  8.3,  3.8, 10. ,
        6.1,  5.8,  3.8,  6.7,  1.7,  8. ,  4.6,  8.2,  1.5,  2.5,  4.4,
        5.8,  8.7,  8.2,  5.9,  5.6,  2.5,  6.1,  2.1,  2.8,  2.5,  1.6,
        7.8,  2.4,  6.1,  8.9,  9. ])
```

Masking

In [24]:
```
x>1
```
Out[24]:
```
array([[ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
        False],
       [ True, False,  True, False,  True,  True,  True,  True,  True,
         True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
        False],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True],
       [ True,  True,  True, False, False,  True,  True,  True,  True,
         True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True],
       [ True,  True, False,  True,  True,  True,  True,  True,  True,
         True]])
```

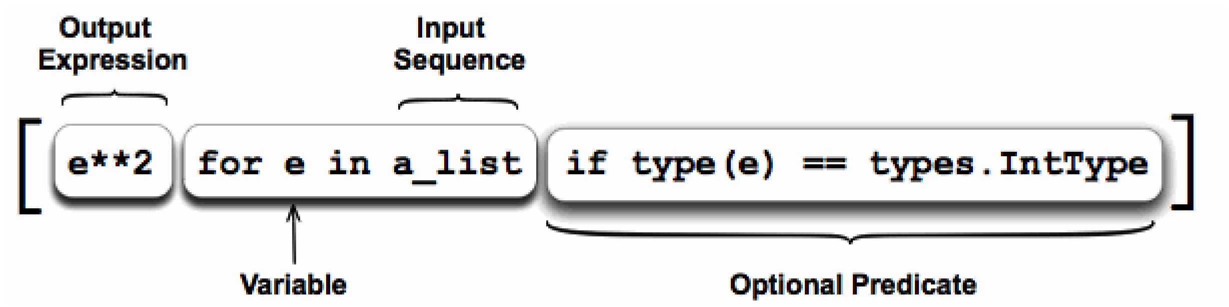
In [27]:
```
Condition = x>1
Condition
x[Condition]
```
Out[27]:
```
array([ 1.2,  2.8,  4.3,  1.8,  8.3,  7.3,  8.9,  4. ,  7.3,  9.9,  2. ,
        7.9,  3.8,  2.6,  2.6,  6.4,  4.6,  4.8,  7.4,  6.2,  2.9,  2.6,
        4.1,  7.3,  8.9,  3.7,  4.9,  8.3,  3. ,  8. ,  4. ,  9.2,  5.8,
        8.6,  5.4,  1.2,  2.5,  9.7,  6.1,  7.5,  6.9,  3.3,  6.1,  8.8,
        6.2,  7.1,  7. ,  4.9,  5.1,  8.1,  2.7,  6.9,  3.1,  1.3,  3.3,
        5.3,  7.6,  3.7,  6.7,  3.7,  8.2,  9.5,  1.1,  8.3,  3.8, 10. ,
        6.1,  5.8,  3.8,  6.7,  1.7,  8. ,  4.6,  8.2,  1.5,  2.5,  4.4,
        5.8,  8.7,  8.2,  5.9,  5.6,  2.5,  6.1,  2.1,  2.8,  2.5,  1.6,
        7.8,  2.4,  6.1,  8.9,  9. ])
```

In [ ]:
```
import numpy as np
a = np.arange(10)
b = a[2:8:2] # [start:end:increment]
print (a)
print (b)
```

In [ ]:
```
a = np.matrix([[1,2,3,4,5], [6,7,8,9,10]])
b = np.array([[1,2,3,4,5], [6,7,8,9,10]])
print(a[0:4,0:4])
print(b[0:4,0:4])
```

List Comprehensions

Picture credit: https://python-3-patterns-idioms-test.readthedocs.io/en/latest/Comprehensions.html (https://python-3-patterns-idioms-test.readthedocs.io/en/latest/Comprehensions.html)



In [28]:
```
data = []
for i in range(11):
     data.append(i)
print(data)

data = [i for i in range(11)]
print(data)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

In [31]:
```
# Another example on masking
import numpy as np
a = np.array([[i for i in range(11)], [i for i in range(11)]],dtype = int)
b = np.array([[True for i in range(11)], [False for i in range(11)]],dtype = bool)
print(a[b])
```

```
[ 0  1  2  3  4  5  6  7  8  9 10]
```

In [32]:
```
# Matrix operations
c=np.matrix([[4, 3], [2, 1]])
d=np.matrix([[1, 2], [3, 4]])
print(np.dot(c,d))
```

```
[[13 20]
 [ 5  8]]
```

Python Lambda

https://www.w3schools.com/python/python_lambda.asp (https://www.w3schools.com/python/python_lambda.asp)

Python map() Function

https://www.w3schools.com/python/ref_func_map.asp (https://www.w3schools.com/python/ref_func_map.asp)

In Class Assignment

Q: Find a short expression to build the matrix:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 9 & 7 & 5 & 3 & 1 & -1 & -3 \\ 4 & 8 & 16 & 32 & 64 & 128 & 256 \end{bmatrix}$$

Recommended tools: List comprehensions, numpy arrays and map().

In [9]:
```
np.array([
    [i for i in range(7)],
    [i for i in range(9,-4,-2)],
    [2**i for i in range(2,9)]
])
```

Out[9]:
```
array([[  0,   1,   2,   3,   4,   5,   6],
       [  9,   7,   5,   3,   1,  -1,  -3],
       [  4,   8,  16,  32,  64, 128, 256]])
```

Q:Given a following 3 x 4 matrix:

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

perform the following operations:

(a) Extract the 3rd column of matrix A and store it in vector B.

In [37]:
```
A= np.matrix([[1,2,3,4],
             [5,6,7,8],
             [9,10,11,12]])
A
```

Out[37]:
```
matrix([[ 1,  2,  3,  4],
        [ 5,  6,  7,  8],
        [ 9, 10, 11, 12]])
```

(b) Extract the 1st and 3rd columns of matrix A and store them in matrix C.

In [74]:
```
C = np.append([A[:,0]],[A[:,2]],axis=2)
C
```

Out[74]:
```
array([[[ 1,  3],
        [ 5,  7],
        [ 9, 11]]])
```

(c) Add the 1st and 3rd rows of matrix A together and store the result in vector D

In [75]:
```
D = A[0,:]+A[2,:]
D
```

Out[75]:
```
matrix([[10, 12, 14, 16]])
```

(d) Change the value in the 2nd row and 3rd column of A to -7 (instead of +7) and call the result AA (do not destroy/change the original A matrix).

In [70]:
```
Anew = A.copy()
Anew[1,2] = -7
Anew
```

Out[70]:
```
matrix([[ 1,  2,  3,  4],
        [ 5,  6, -7,  8],
        [ 9, 10, 11, 12]])
```