Data Science & Data Analytics Laboratory

References:

1. Learning Python, 5th Edition by Mark Lutz
2. Python in a Nutshell, 3rd Edition by Alex Martelli, Anna Ravenscroft & Steve Holden
3. Core Python Programming by N. Rao
4. Programming in Python 3 by Mark Summerfield
5. Python for Everybody by Charles Severance

6. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition by Aurélien Géron
7. Python for Data Analysis, 2nd Edition by William McKinney
8. Python for Data Science for Dummies, 2ed by Mueller

In [2]:

```
import numpy as np
```

In [3]:

```
array = np.arange(20)
print(type(array))
print(array)
```

```
<class 'numpy.ndarray'>
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
```

In [5]:

```
print(array.shape)
print(type(array.shape))
```

```
(20,)
<class 'tuple'>
```

In [6]:

```
array[3]
```

Out[6]:

```
3
```

In [7]:

```
#mutable
array[3] = 100
print(array)
```

```
[  0   1   2 100   4   5   6   7   8   9  10  11  12  13  14  15  16  17
  18  19]
```

In [13]:

```
array = np.arange(9)
print(array)
x = array.reshape(3,3)
print(x)
```

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

In [10]:

```
np.arange(0,10)
```

Out[10]:

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [11]:

```
np.arange(10, 35, 3)
```

Out[11]:

```
array([10, 13, 16, 19, 22, 25, 28, 31, 34])
```

In [14]:

```
np.zeros((2,4))
```

Out[14]:

```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

In [15]:

```
np.ones((3,4))
```

Out[15]:

```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

In [16]:

```
np.full((2,2), 3)
```

Out[16]:

```
array([[3, 3],
       [3, 3]])
```

In [17]:

```
np.eye(3,3)
```

Out[17]:

```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

In [18]:

```
my_list = [1,2,3,4,5,6,7,8]
my_array = np.array(my_list)
print(my_array)
print(type(my_array))
```

```
[1 2 3 4 5 6 7 8]
<class 'numpy.ndarray'>
```

In [19]:

```
my_array = my_array.reshape(2,4)
print(my_array)
```

```
[[1 2 3 4]
 [5 6 7 8]]
```

In [20]:

```
my_array = my_array.T
print(my_array)
```

```
[[1 5]
 [2 6]
 [3 7]
 [4 8]]
```

In [21]:

```
max = my_array.max()
min = my_array.min()
mean = my_array.mean()
std = my_array.std(axis = 1)
print("Max : ",max)
print("Min : ",min)
print("Mean : ",mean)
print("Standard Deviation : ",std)
```

```
Max :  8
Min :  1
Mean :  4.5
Standard Deviation :  [2. 2. 2. 2.]
```

In [22]:

```
num = []
for i in range(0,5):
    num.append(np.random.randint(0,2))
num = np.array(num)
print(num)
print(np.unique(num))
```

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

In [ ]:

```
x = np.arange(1,4)
y = np.arange(1,7,2)
print(x)
print(y)
np.add(x,y)
```

In [ ]:

```
num = np.arange(1,10,dtype=float).reshape(3,3)
print(num)
print(np.max(num))
print(np.max(num,axis = 0))
print(np.max(num,axis = 1))
```

In [ ]:

```
num[1,2] = np.NaN
print(num)
np.max(num,axis = 0)
```

In [ ]:

```
import pandas as pd
```

In [ ]:

```
data = pd.read_excel('test_data.xlsx')
```

In [ ]:

```
data.head()
```

In [ ]:

```
data.isnull().sum()
```

In [ ]:

```
data.dropna()
```

In [ ]:

```
data.dropna(axis = 1)
```

In [ ]:

```
import numpy as np
from sklearn.impute import SimpleImputer
imr = SimpleImputer(missing_values=np.nan, strategy='mean')
imr = imr.fit(data)
imputed_data = imr.transform(data)
print(data)
print(imputed_data)
```

In [ ]:

```
data = pd.read_csv('iris.csv')
data.head()
```

In [ ]:

```
data = pd.read_csv('iris.csv',header = None)
data.head()
```

In [ ]:

```
data.columns = ['sepal length','sepal width','petal length','petal width','class']
data.head()
```

In [ ]:

```
np.unique(data['class'])
```

In [ ]:

```
mapping = {'Iris-setosa':0,
'Iris-versicolor':1,
'Iris-virginica':2}
data['class'] = data['class'].map(mapping)
data.head()
```

In [ ]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['class'] = le.fit_transform(data['class'])
data.head()
```