numpy practice session

In [1]:

```
#pip install numpy
```

In [2]:

```
# import this library in jypyter notebook
import numpy as np
```

Creating an array using numpy

In [3]:

```
# 1-D array
food = np.array(["Pakora", "Samosa", "Raita"])
food
```

Out[3]:

```
array(['Pakora', 'Samosa', 'Raita'], dtype='<U6')
```

In [4]:

```
price = np.array([5,5,5])
price
```

Out[4]:

```
array([5, 5, 5])
```

In [5]:

```
type(price)
```

Out[5]:

```
numpy.ndarray
```

In [6]:

```
type(food)
```

Out[6]:

```
numpy.ndarray
```

In [7]:

```
len(price)
```

Out[7]:

```
3
```

In [8]:

```
len(food)
```

Out[8]:

```
3
```

In [9]:

```
price[0:]
```

Out[9]:

```
array([5, 5, 5])
```

In [10]:

```
food[1]
```

```
'Samosa'
```

Out[10]:

In [11]:

```
price.mean()
```

Out[11]:

```
5.0
```

In [12]:

```
# zeros method
np.zeros(6)
```

Out[12]:

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

In [13]:

```
# ones
np.ones(5)
```

Out[13]:

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

In [14]:

```
# empty
np.empty(5)
```

Out[14]:

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

In [15]:

```
# range
np.arange(10)
```

Out[15]:

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

In [16]:

```
# specify
np.arange(2,20)
```

Out[16]:

```
array([ 2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19])
```

In [17]:

```
# specify interval
np.arange(2,20, 2)
```

Out[17]:

```
array([ 2,  4,  6,  8, 10, 12, 14, 16, 18])
```

In [18]:

```
# table
np.arange(0,55,5)
```

Out[18]:

```
array([ 0,  5, 10, 15, 20, 25, 30, 35, 40, 45, 50])
```

In [19]:

```
# line space
np.linspace(1, 100, num=10)
```

Out[19]:

```
array([  1.,  12.,  23.,  34.,  45.,  56.,  67.,  78.,  89., 100.])
```

In [20]:
```
# specify your data type
np.ones(50, dtype=np.int64)
```

Out[20]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1], dtype=int64)
```

In [21]:
```
np.ones(50, dtype=np.float64)
```

Out[21]:
```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```

Array functions

In [22]:
```
a= np.array([10,12,15,2,4,6,100,320,0.5,10,3])
a
```

Out[22]:
```
array([ 10. ,  12. ,  15. ,   2. ,   4. ,   6. , 100. , 320. ,   0.5,
        10. ,   3. ])
```

In [23]:
```
a.sort()
a
```

Out[23]:
```
array([  0.5,   2. ,   3. ,   4. ,   6. ,  10. ,  10. ,  12. ,  15. ,
       100. , 320. ])
```

In [24]:
```
b= np.array([10.2,3.4,53.6,91.6,45.5])
b
```

Out[24]:
```
array([10.2,  3.4, 53.6, 91.6, 45.5])
```

In [25]:
```
c= np.concatenate((a,b))
c
```

Out[25]:
```
array([  0.5,   2. ,   3. ,   4. ,   6. ,  10. ,  10. ,  12. ,  15. ,
       100. , 320. ,  10.2,   3.4,  53.6,  91.6,  45.5])
```

In [26]:
```
c.sort()
c
```

Out[26]:
```
array([  0.5,   2. ,   3. ,   3.4,   4. ,   6. ,  10. ,  10. ,  10.2,
        12. ,  15. ,  45.5,  53.6,  91.6, 100. , 320. ])
```

2-D arrays

In [27]:
```
a= np.array([[1,2], [5,4]])
a
```

Out[27]:
```
array([[1, 2],
       [5, 4]])
```

In [28]:

```
b= np.array([[6,7],[8,9]])
b
```

Out[28]:

```
array([[6, 7],
       [8, 9]])
```

In [29]:

```
np.concatenate((a,b), axis=0)
```

Out[29]:

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

In [30]:

```
np.concatenate((a,b), axis=1)
```

Out[30]:

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

In [31]:

```
a= np.array([[[0,1,2,3],
                      [4,5,6,7]],

             [[0,1,2,3],
             [4,5,6,7]],

             [[0,1,2,3],
             [4,5,6,7]]])
a
```

Out[31]:

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

       [[0, 1, 2, 3],
        [4, 5, 6, 7]],

       [[0, 1, 2, 3],
        [4, 5, 6, 7]]])
```

In [32]:

```
# to find the number  of dimesions
a.ndim
```

Out[32]:

```
3
```

In [33]:

```
b= np.array([[5,6,7],
           [8,9,10],
           [10,11,12]])
b
```

Out[33]:

```
array([[ 5,  6,  7],
       [ 8,  9, 10],
       [10, 11, 12]])
```

In [34]:

```
b.ndim
```

Out[34]:

```
2
```

In [35]:

```
#size(number of elements)
a.size
```

Out[35]:

```
24
```

In [36]:

```
#shape
a.shape
```

Out[36]:

```
(3, 2, 4)
```

In [37]:

```
a= np.arange(9) #3*3
a
```

Out[37]:

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

In [38]:

```
#reshape
a= a.reshape(3,3) #3*3=9
a
```

Out[38]:

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

In [39]:

```
# reshape
np.reshape(a, newshape=(1,9), order='c')
```

Out[39]:

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

Convert 1-D into 2-D array

In [40]:

```
a =np.array([1,2,3,4,5,6,7,8,9])
a
```

Out[40]:

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

In [41]:

```
a.shape
```

Out[41]:

```
(9,)
```

In [42]:

```
# row wise 2D conversion
b= a[np.newaxis,:]
b
```

Out[42]:

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

In [43]:

```
b.shape
```

Out[43]:

```
(1, 9)
```

In [44]:

```
# column wice 2D conversion
c= a[:, np.newaxis]
c
```

Out[44]:

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

Indexing or slicing

In [45]:

```
a
```

Out[45]:

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

In [46]:

```
a[2:9]
```

Out[46]:

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

In [47]:

```
a*6
```

Out[47]:

```
array([ 6, 12, 18, 24, 30, 36, 42, 48, 54])
```

In [48]:

```
a+6
```

Out[48]:

```
array([ 7,  8,  9, 10, 11, 12, 13, 14, 15])
```

In [49]:

```
a.sum()
```

Out[49]:

```
45
```

In [50]:

```
a.mean()
```

Out[50]:

```
5.0
```