numpy practice session

pip install numpy

In [1]:

```
# import this Library in j.notebook
import numpy as np
```

creating an array using numpy

In [2]:

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

Out[2]:

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

numbers key

In [3]:

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

Out[3]:

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

In [4]:

```
type(price)
```

Out[4]:

```
numpy.ndarray
```

In [5]:

```
type(food)
```

Out[5]:

```
numpy.ndarray
```

length

In [6]:

```
len(food)
```

Out[6]:

```
3
```

In [7]:

```
len(price)
```

Out[7]:

```
3
```

Indexing

In [8]:

```
price[1]
```

Out[8]:

```
5
```

In [9]:

```
price[0:]
```

Out[9]:

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

In [10]:

```
food[0]
```

Out[10]:

```
'Pakora'
```

In [11]:

```
food[2]
```

Out[11]:

```
'Raita'
```

In [12]:

```
food[1]
```

Out[12]:

```
'Samosa'
```

In [13]:

```
food[0:]
```

Out[13]:

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

In [14]:

```
price.mean()
```

Out[14]:

```
5.0
```

zeros

In [15]:

```
np.zeros(9)
```

Out[15]:

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

ones

In [ ]:

```
```

In [16]:

```
np.ones(9)
```

Out[16]:

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

empty

In [17]:

```
x=np.empty(5)
x
```

Out[17]:
```
array([0., 0., 0., 0., 0.])
```

In [18]:
```
for i in range(5):
    x[i]=i
x
```

Out[18]:
```
array([0., 1., 2., 3., 4.])
```

Range

In [19]:
```
np.arange(10)
```

Out[19]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [20]:
```
#specify
np.arange(2,10)
```

Out[20]:
```
array([2, 3, 4, 5, 6, 7, 8, 9])
```

specific interval

In [21]:
```
np.arange(2,20,2)
```

Out[21]:
```
array([ 2,  4,  6,  8, 10, 12, 14, 16, 18])
```

table of 5

In [22]:
```
np.arange(5,55,5)
```

Out[22]:
```
array([ 5, 10, 15, 20, 25, 30, 35, 40, 45, 50])
```

specific line space

In [23]:
```
np.linspace(0,10,num=5)
```

Out[23]:
```
array([ 0. ,  2.5,  5. ,  7.5, 10. ])
```

specify your data type

In [24]:
```
np.ones(50,dtype=np.int64)
```

Out[24]:
```
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 [25]:
```
np.ones(50,dtype=np.float64)
```

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
```

Out[25]:

```
       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 [26]:

```
a=np.array([10,12,15,6,100,3,320,0.5,10.3])
a
```

Out[26]:

```
array([ 10. ,  12. ,  15. ,   6. , 100. ,   3. , 320. ,   0.5,  10.3])
```

In [27]:

```
a.sort()
a
```

Out[27]:

```
array([  0.5,   3. ,   6. ,  10. ,  10.3,  12. ,  15. , 100. , 320. ])
```

In [28]:

```
b=np.array([10.2,3.4,53.6,76.5])
b
```

Out[28]:

```
array([10.2,  3.4, 53.6, 76.5])
```

In [29]:

```
c=np.concatenate((a,b))
c
```

Out[29]:

```
array([  0.5,   3. ,   6. ,  10. ,  10.3,  12. ,  15. , 100. , 320. ,
        10.2,   3.4,  53.6,  76.5])
```

In [30]:

```
c.sort()
c
```

Out[30]:

```
array([  0.5,   3. ,   3.4,   6. ,  10. ,  10.2,  10.3,  12. ,  15. ,
        53.6,  76.5, 100. , 320. ])
```

2-D array

In [31]:

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

Out[31]:

```
array([[1, 2],
       [5, 4]])
```

In [32]:

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

Out[32]:

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

In [33]:

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

Out[33]:

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

In [34]:

```
c.ndim
```

Out[34]:

```
2
```

In [35]:

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

Out[35]:

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

In [36]:

```
d.ndim
```

Out[36]:

```
2
```

3-D array

In [37]:

```
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[37]:

```
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]]])
```

to find the number of dimensions

In [38]:

```
a.ndim
```

Out[38]:

```
3
```

In [39]:

```
type(a)
```

Out[39]:

```
numpy.ndarray
```

In [40]:

```
b=np.array([[5,6,7],
            [8,9,10],
            [11,12,13]])
b
```

Out[40]:
```
array([[ 5,  6,  7],
       [ 8,  9, 10],
       [11, 12, 13]])
```

In [41]:
```
b.ndim
```

Out[41]:
```
2
```

size (number of elements)

In [42]:
```
a.size
```

Out[42]:
```
24
```

In [43]:
```
b.size
```

Out[43]:
```
9
```

shape

In [44]:
```
a.shape
```

Out[44]:
```
(3, 2, 4)
```

In [45]:
```
b.shape
```

Out[45]:
```
(3, 3)
```

Reshape

In [46]:
```
a=np.arange(9)
a
```

Out[46]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8])
```

In [47]:
```
b=a.reshape(3,3)
b
```

Out[47]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

In [48]:
```
# reshape
np.reshape(a,newshape=(9),order="c")
```

Out[48]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8])
```

convert 1-D into 2-D

In [49]:

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

Out[49]:

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

In [50]:

```
a.shape
```

Out[50]:

```
(9,)
```

In [51]:

```
b=a[np.newaxis,:]
b
```

Out[51]:

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

In [52]:

```
b.shape
```

Out[52]:

```
(1, 9)
```

In [53]:

```
c=a[:, np.newaxis]
c
```

Out[53]:

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

Indexing and slicing

In [54]:

```
a
```

Out[54]:

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

In [55]:

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

Out[55]:

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

multiplication

In [56]:

```
a*6
```

Out[56]:

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

concatenating

In [57]:

```
a+6
```

Out[57]:

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

In [58]:

```
a.sum()
```

Out[58]:

```
45
```

In [59]:

```
a.mean()
```

Out[59]:

```
5.0
```

In [60]:

```
a.max()
```

Out[60]:

```
9
```

In [61]:

```
a.min()
```

Out[61]:

```
1
```

In [62]:

```
y=np.empty(11)
y
```

Out[62]:

```
array([1.72469605e-312, 2.07507571e-322, 0.00000000e+000, 0.00000000e+000,
       8.01097888e-307, 1.16095484e-028, 9.46258956e-076, 6.32672800e+180,
       4.74483502e+170, 4.59210323e-072, 7.22594635e+159])
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: