numpy practice session

In [ ]:
```
# pip install numpy
```

In [1]:
```
# import this library in j.notebook
import numpy as np
```

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

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

In [6]:
```
type(price)
```

Out[6]:
```
numpy.ndarray
```

In [7]:
```
type(food)
```

Out[7]:
```
numpy.ndarray
```

In [8]:
```
len(food)
```

Out[8]:
```
3
```

In [9]:
```
len(price)
```

Out[9]:
```
3
```

In [11]:
```
price[0:]
```

Out[11]:
```
array([5, 5, 5])
```

In [12]:
```
food[1]
```

Out[12]:
```
'samosa'
```

In [14]:
```
price.mean()
```

Out[14]:
```
5.0
```

In [15]:
```
#zeros method to make array
np.zeros(6)
```

Out[15]:
```
array([0., 0., 0., 0., 0., 0.])
```

In [16]:
```
# ones
np.ones(5)
```

Out[16]:
```
array([1., 1., 1., 1., 1.])
```

In [17]:
```
# empty
np.empty(5)
```

Out[17]:
```
array([1., 1., 1., 1., 1.])
```

In [ ]:
```
# range
```

In [18]:
```
np.arange(10)
```

Out[18]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

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

Out[20]:
```
array([ 2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19])
```

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

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

In [23]:
```
# table
np.arange(5,55,5)
```

Out[23]:
```
array([ 5, 10, 15, 20, 25, 30, 35, 40, 45, 50])
```

In [24]:
```
# line space
np.linspace(0,10,num=5)
```

Out[24]:
```
array([ 0. ,  2.5,  5. ,  7.5, 10. ])
```

In [25]:
```
# specify your data type
np.ones(50,dtype=np.int64)
```

Out[25]:
```
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], dtype=int64)
```

In [27]:
```
np.ones(50,dtype=np.float64)
```

Out[27]:
```
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.])
```

Array function

In [28]:
```
a=np.array([4,5,6,7,12,12.3,13.6,17.2])
a
```

Out[28]:
```
array([ 4. ,  5. ,  6. ,  7. , 12. , 12.3, 13.6, 17.2])
```

In [31]:
```
a.sort()
a
```

Out[31]:
```
array([ 4. ,  5. ,  6. ,  7. , 12. , 12.3, 13.6, 17.2])
```

In [32]:
```
b=np.array([5,6,4,3.4,2.4,5.8])
b
```

Out[32]:
```
array([5. , 6. , 4. , 3.4, 2.4, 5.8])
```

In [34]:
```
c=np.concatenate((a,b))
c
```

Out[34]:
```
array([ 4. ,  5. ,  6. ,  7. , 12. , 12.3, 13.6, 17.2,  5. ,  6. ,  4. ,
        3.4,  2.4,  5.8])
```

In [35]:
```
c.sort()
c
```

Out[35]:
```
array([ 2.4,  3.4,  4. ,  4. ,  5. ,  5. ,  5.8,  6. ,  6. ,  7. , 12. ,
       12.3, 13.6, 17.2])
```

In [ ]:
```

```

2 D arrays

In [40]:
```
a=np.array([[1,2],[5,4]])
a
```

Out[40]:
```
array([[1, 2],
       [5, 4]])
```

In [38]:
```
b=np.array([[6,7],[7,8]])
b
```

Out[38]:
```
array([[6, 7],
       [7, 8]])
```

In [45]:
```
np.concatenate((a,b), axis=0)
```

Out[45]:
```
array([[1, 2],
       [5, 4],
       [6, 7],
       [7, 8]])
```

In [46]:
```
np.concatenate((a,b), axis=1)
```

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

In [47]:
```
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[47]:
```
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 [49]:
```
# to find the number of dimension
a.ndim
```

Out[49]:
```
3
```

In [56]:
```
b=np.array([[5,6,7],
            [8,9,10],
            [10,11,12]])
b
```

Out[56]:
```
array([[ 5,  6,  7],
       [ 8,  9, 10],
       [10, 11, 12]])
```

In [57]:
```
b.ndim
```

Out[57]:
```
2
```

In [58]:
```
type(a)
```

Out[58]:
```
numpy.ndarray
```

In [59]:
```
type(b)
```

Out[59]:
```
numpy.ndarray
```

In [62]:
```
b.size
```

Out[62]:
```
9
```

In [63]:
```
a.size
```

Out[63]:
```
2
```

In [67]:
```
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[67]:
```
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 [68]:
```
# size (no of element)
a.size
```

Out[68]:
```
24
```

In [70]:
```
# shape
a.shape
```

Out[70]:
```
(3, 2, 4)
```

In [71]:
```
a=np.arange(9)
a
```

Out[71]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8])
```

In [72]:
```
# reshape
b=a.reshape(3,3)
b
```

Out[72]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

In [75]:
```
# convert 1-D into 2-D array
a=np.array([1,2,3,4,5,6,7,8,9])
a
```

Out[75]:
```
array([1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [76]:
```
a.shape
```

Out[76]:
```
(9,)
```

In [78]:
```
# row waise conversion
b=a[np.newaxis, :]
b
```

Out[78]:
```
array([[1, 2, 3, 4, 5, 6, 7, 8, 9]])
```

In [79]:
```
b.shape
```

Out[79]:
```
(1, 9)
```

In [80]:
```
#coloumn waise conversion
c=a[: ,np.newaxis]
c
```

Out[80]:
```
array([[1],
       [2],
       [3],
       [4],
       [5],
       [6],
       [7],
       [8],
       [9]])
```

In [81]:
```
a
```

Out[81]:
```
array([1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [83]:
```
a[2:9]
```

Out[83]:
```
array([3, 4, 5, 6, 7, 8, 9])
```

In [84]:
```
a*6
```

Out[84]:
```
array([ 6, 12, 18, 24, 30, 36, 42, 48, 54])
```

In [85]:
```
a+6
```

Out[85]:
```
array([ 7,  8,  9, 10, 11, 12, 13, 14, 15])
```

In [86]:
```
a.sum()
```

Out[86]:
```
45
```

In [87]:
```
a.mean()
```

Out[87]:
```
5.0
```

In [ ]:
```
# link
# numpy then user guid
```

In [ ]:

In [ ]:

In [ ]: