Numpy practice

In [ ]:

```
#pip install numpy
```

In [1]:

```
# import numpy
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Creating an array using numpy

In [2]:

```
#1d array
food = np.array(["pakora","samosa","raita"])
food
```

Out[2]:

```
array(['pakora', 'samosa', 'raita'], dtype='<U6')
```

In [3]:

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

Out[3]:

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

In [4]:

```
type(food)
```

Out[4]:

```
numpy.ndarray
```

In [5]:

```
len(food)
```

Out[5]:

```
3
```

In [6]:

```
price[0:3]
```

Out[6]:

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

In [7]:

```
food[2]
```

Out[7]:

```
'raita'
```

In [8]:

```
price.mean() # find mean
```

Out[8]:

```
5.0
```

In [9]:

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

Out[9]:

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

In [10]:

```
np.ones(6)  # for all one array
```

Out[10]:

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

In [11]:

```
# assignment search on google what the answer here
# empty
np.empty(6)  # its direct belong to ones method
```

Out[11]:

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

In [12]:

```
np.arange(23)  # this duction show the numbers upto  which you pass.
```

Out[12]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22])
```

In [13]:

```
# specify
np.arange(2,56)
```

```
Out[13]: array([ 2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
                19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
                36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52,
                53, 54, 55])
```

```
In [14]: # specify intervel
         np.arange(2,56,4)
```

```
Out[14]: array([ 2,  6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54])
```

```
In [15]: # through this we make table
         # like table of 5
         np.arange(5,55,5)
```

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

```
 In [ ]: # line space  thorugh this we create randon data like range data
         np.linspace(1,100,num=50)
```

```
In [16]: # specify your data typr
         np.ones(34, dtype=np.int64)
```

```
Out[16]: 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], dtype=int64)
```

```
In [17]: # specify your data typr
         np.ones(34, dtype=np.float64)
```

```
Out[17]: 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.])
```

```
In [18]: # specify your data typr
         np.ones('r', dtype=np.char64)  # errror no run
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-18-aef344894925> in <module>
```

```
      1 # specify your data typr
----> 2 np.ones('r', dtype=np.char64)  # errror no run

~\anaconda3\lib\site-packages\numpy\__init__.py in __getattr__(attr)
    301                 return Tester
    302
--> 303             raise AttributeError("module {!r} has no attribute "
    304                                  "{!r}".format(__name__, attr))
    305

AttributeError: module 'numpy' has no attribute 'char64'
```

In [19]:

```
# specify your data typr
np.ones(34, dtype=np.float32)
# we use for the if we have out of stock value come
```

Out[19]:

```
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.],
      dtype=float32)
```

ARRAY function

In [20]:

```
a = np.array([12,45,65,744,5.5,67.6])
a
```

Out[20]:

```
array([ 12. ,  45. ,  65. , 744. ,   5.5,  67.6])
```

In [21]:

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

Out[21]:

```
array([  5.5,  12. ,  45. ,  65. ,  67.6, 744. ])
```

In [22]:

```
#you can practice all function just click on tab`
```

In [23]:

```
b= np.array([3,4,5.5,6,8,3.8])
b
```

Out[23]:

```
array([3. , 4. , 5.5, 6. , 8. , 3.8])
```

```
In [24]: c=np.concatenate((a,b))
         c

Out[24]: array([ 12. ,  45. ,  65. , 744. ,   5.5,  67.6,   3. ,   4. ,   5.5,
           6. ,   8. ,   3.8])

In [25]: c.sort()
         c

Out[25]: array([  3. ,   3.8,   4. ,   5.5,   5.5,   6. ,   8. ,  12. ,  45. ,
          65. ,  67.6, 744. ])
```

2d array

```
In [26]: a= np.array([[1,2,3,4],[4,46,66,6]])
         a

Out[26]: array([[ 1,  2,  3,  4],
                [ 4, 46, 66,  6]])

In [ ]: 

In [27]: b=np.array([[3,5,6],[4,5,6]])
         b

Out[27]: array([[3, 5, 6],
                [4, 5, 6]])

In [28]: c =np.concatenate((a,b))
         c

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-28-ad7468735828> in <module>
----> 1 c =np.concatenate((a,b))
      2 c
```

```
<__array_function__ internals> in concatenate(*args, **kwargs)

ValueError: all the input array dimensions for the concatenation axis must match exactly, but along dimension 1, the array at inde
x 0 has size 4 and the array at index 1 has size 3
```

In [ ]:

```
```

In [29]:

```
# now  the 2d array in one frame # it will add becaue it size same a and b-
c= np.array((a,b))
c
```

```
<ipython-input-29-39b216b5cdc5>:2: VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences (which is a list-or
-tuple of lists-or-tuples-or ndarrays with different lengths or shapes) is deprecated. If you meant to do this, you must specify
'dtype=object' when creating the ndarray.
  c= np.array((a,b))
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-29-39b216b5cdc5> in <module>
      1 # now  the 2d array in one frame # it will add becaue it size same a and b-
----> 2 c= np.array((a,b))
      3 c

ValueError: could not broadcast input array from shape (2,4) into shape (2,)
```

- if a and b array dimension are equal then it wil run and add them.
- if a and b array index are not equal then it come error
- if dimension are not equal then add them ? if :yes. Assignment

In [34]:

```
# now ADD THEM
a= np.array([[1,2,4],[4,4,6]])
a
```

Out[34]:

```
array([[1, 2, 4],
       [4, 4, 6]])
```

In [33]:

```
b= np.array([[3,4,5],[5,6,3.8]])
b
```

Out[33]:
```
array([[3. , 4. , 5. ],
       [5. , 6. , 3.8]])
```

In [36]:
```
# now add them
c=np.array((a,b))
c
```

Out[36]:
```
array([[[1. , 2. , 4. ],
        [4. , 4. , 6. ]],

       [[3. , 4. , 5. ],
        [5. , 6. , 3.8]]])
```

In [37]:
```
c=np.concatenate((a,b))
c
```

Out[37]:
```
array([[1. , 2. , 4. ],
       [4. , 4. , 6. ],
       [3. , 4. , 5. ],
       [5. , 6. , 3.8]])
```

In [38]:
```
# axis =0   aslo called stack
c=np.concatenate((a,b),axis=0)
c
```

Out[38]:
```
array([[1. , 2. , 4. ],
       [4. , 4. , 6. ],
       [3. , 4. , 5. ],
       [5. , 6. , 3.8]])
```

In [39]:
```
# axis = 1  aslo called stack
c=np.concatenate((a,b),axis=1)
c
```

Out[39]:
```
array([[1. , 2. , 4. , 3. , 4. , 5. ],
       [4. , 4. , 6. , 5. , 6. , 3.8]])
```

3d Array

In [40]:

```
# def:
we need three 2d  dimension array then maked 3d array
arr1 = np.array([[[2,17], [45, 78]], [[88, 92], [60, 76]],[[76,33],[20,18]]])
arr1
```

Out[40]:

```
array([[[ 2, 17],
        [45, 78]],

       [[88, 92],
        [60, 76]],

       [[76, 33],
        [20, 18]]])
```

In [41]:

```
# find  the dimension numbers
arr1.ndim
```

Out[41]:

```
3
```

In [43]:

```
b=np.array([[2,4,5],
           [2,5,6,],
           [3,4,5]])
b
```

Out[43]:

```
array([[2, 4, 5],
       [2, 5, 6],
       [3, 4, 5]])
```

In [44]:

```
b.ndim # it mean this is 2d array
```

Out[44]:

```
2
```

In [45]:

```
# SIZE
arr1.size
```

Out[45]:

```
12
```

In [47]:

```
# shape   it mean  3 maen 3d and 2row and 2 colom matrix
```

```
arr1.shape
```

Out[47]:

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

In [52]:

```
arr1=np.arange(9) # 3*3=9
arr1
```

Out[52]:

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

In [54]:

```
# reshape
# thorough this we cahnge the big into row and colom
b= arr1.reshape(3,3)  #3*3=9
b
```

Out[54]:

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

In [57]:

```
# reshape newshape
# thorough this we cahnge the big into row and colom
np.reshape(arr1,newshape=(1,9),order='C')   # C mean change into colom
```

Out[57]:

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

In [59]:

```
# comvert into 1 day into 2 d
a=np.array([0, 1, 2, 3, 4, 5, 6, 7, 8])
a
```

Out[59]:

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

In [60]:

```
# first chek shape  # answer is (9 mean 9 element have )
a.shape
```

Out[60]:

```
(9,)
```

In [62]:

```
# change into row
```

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

Out[62]:

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

In [63]:

```
b.shape  # you can any typ conversion like row into colom or colom into row
```

Out[63]:

```
(1, 9)
```

In [65]:

```
# change into colom
b = a[: ,np.newaxis]
b
```

Out[65]:

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

In [66]:

```
# slicing is same
a[::]
a
```

Out[66]:

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

In [72]:

```
a[2:8] # some time last item exclusive or some time exclusive
```

Out[72]:

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

In [74]:

```
#repition
a*2
```

Out[74]:

```
array([ 0,  2,  4,  6,  8, 10, 12, 14, 16])
```

```
In [75]: # addition
         a+4
```

```
Out[75]: array([ 4,  5,  6,  7,  8,  9, 10, 11, 12])
```

```
In [77]: # sum function
         a.sum()
```

```
Out[77]: 36
```

```
In [78]: # array mean
         a.mean()
```

```
Out[78]: 4.0
```

```
In [ ]: # go to numpy --> numpy user guide -->absoulte begneer guide--> then every thin here
        # practice then make notebook and save
```

```
In [ ]:
```