```
In [ ]: import numpy as np
```

Why numpy(Numerical Python)? :- As a we know list is an alternative of array. But when data science introduced then large number of data is being stored in list but time complexity of list is very high because list use dynamic and referential array concept.That's why numpy is introduced to solve this problem because it is memory efficient,used for working with array.A Numpy array is like special list in python with same data type.

So,we will create Array and deal with array in numpy.

```
In [ ]: # creation of an araay
        arr = np.array([25,41,63,66,85,74])
        arr

Out[ ]: array([25, 41, 63, 66, 85, 74])

In [ ]: type(arr)
        # ndarray -> n dimensional array

Out[ ]: numpy.ndarray

In [ ]: arr.size  #no. of item

Out[ ]: 6

In [ ]: len(arr)

Out[ ]: 6

In [ ]: #  to check the dimension of array
        arr.ndim

Out[ ]: 1

In [ ]: # to check data type of array
        arr.dtype

Out[ ]: dtype('int64')

In [ ]: print(np.__version__)

       2.0.0

In [ ]: arr2 =np.array([25,41,63,66,85,74,'upflairs',True])  # make all data into string
        arr2

Out[ ]: array(['25', '41', '63', '66', '85', '74', 'upflairs', 'True'],
              dtype='<U21')

In [ ]: arr2.dtype

Out[ ]: dtype('<U21')
```

In [ ]:
```
print(arr2)
```

```
['25' '41' '63' '66' '85' '74' 'upflairs' 'True']
```

Indexing and slicing in this same as array as we learn earliar

In [ ]:
```
# slicing is same as previous

print(arr[3])
print(arr[-1])
print(arr[2:5])
print(arr[:6]) # print from start
print(arr[2:])  # print till end
print(arr[2::2])  # jump is 2
```

```
66
74
[63 66 85]
[25 41 63 66 85 74]
[63 66 85 74]
[63 85]
```

In [ ]:
```
arr[-1]=53
```

In [ ]:
```
arr + 2 # add 2 in every element
```

Out[ ]:
```
array([27, 43, 65, 68, 87, 55])
```

Now let's consider 2 dimensional array(arrays under an array).It means there are several single dimension array in an array.

In [ ]:
```
ls = [[1,2,3],[4,5,6],[7,8,9]]
type(ls)
print(ls[0][2])
```

```
3
```

In [ ]:
```
arr3 = np.array(ls)  # here we convert list into array
print(arr3)
print(arr3.ndim)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
2
```

In [ ]:
```
# find no. of rows and column
arr3.shape  #(row,column)
```

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

Excessing in 2 dimensional array.

In [ ]:
```
print(arr3[0])  #excessing single row
print(arr3[0][1])  #excessing a single element
print(arr3[2][2])

# excessing multiple rows
arr3[0:2]  # it will show 0th and 1st index row
```

```
# excessing multiple elements
# arr[row,column] -> [row start:row end,col start : col end]
print(arr[1:,1:])   # print 5,6,8,9
```

```
[1 2 3]
2
9
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[19], line 10
      6 arr3[0:2]  # it will show 0th and 1st index row
      8 # excessing multiple elements
      9 # arr[row,column] -> [row start:row end,col start : col end]
---> 10 print(arr[1:,1:])   # print 5,6,8,9

IndexError: too many indices for array: array is 1-dimensional, but 2 were indexe
d
```

In [ ]:
```
arr4 = np.random.randint(1,200,500)   #it will instant make array of 500 element
arr4
```

```
Out[ ]: array([160,   1, 144,  27,  65,  49, 157, 105, 145, 109, 127, 169,  50,
                34,  11,  77,  99, 147, 135, 175,  16, 127,  42,   8,   9,   3,
               189, 162,  98, 185,  61, 151, 115,  40,  48, 127, 193, 166, 158,
                58,  26,  85,  55,  81, 140,   3, 106,  32, 184, 199,  51,  63,
               194, 108,  94, 100, 115, 196, 119,  51, 197, 158, 138, 102, 181,
               106,  42, 193,  33,  71,  73, 102,  69, 167, 127, 164, 188,  23,
               122,  64,  79, 173, 187, 107,   6, 194,  19,   1,  49, 144,  98,
               173,  65, 115,  12, 155,  92,  82,   7,  75,  71,   1, 193,  75,
                96, 124, 178, 129,  21,  13, 177, 112, 176,  92,  94,  11, 167,
               144, 144,  53,  11,  16,  71, 108,  80,  99, 170, 125,  97, 139,
               131,  92,  87, 180, 105,  39,  93,  21, 116, 176,  49,  15, 195,
                51, 112, 101,  88, 106, 196, 181,  30,   1, 140, 139,  67, 199,
               181, 128, 191, 110, 161, 166,  30, 155, 127, 138,  50,  36,  13,
               153,  61,  66,  15,  75, 176,  40,  26,  71,  67,   2, 195, 156,
               104,  88, 199, 117, 179,   6,  94, 168, 130, 132,  97,  53,  38,
               108,  26,  67, 153,  33, 112,  43, 108, 159,  56,   2, 158,  33,
               149, 184,  68, 132,  63,  56, 142, 165,   1,  91, 121, 172,  38,
               188,  52,  15, 191,  31,  14, 158,  42,  29,  75, 101, 115, 164,
               137, 165,  33,   3, 128, 184,  63,  32, 104,   9, 150, 121, 112,
               166,  24, 104,  23, 111,  98, 164,  48,  96, 146, 111,  34,  46,
                29, 144,  67, 110,   5, 193,  18, 118, 199, 137,  51, 193, 194,
                57,  52,  81,  52,  28, 129,  65,  60,  52, 140, 116, 146,  17,
                96, 190,  36, 141, 145,  54, 163, 155,  33, 100,  53, 104, 197,
                19,  40,  65, 164,  77, 102, 170, 157, 177, 128,  13, 176,  22,
                34, 139, 156, 116, 111, 124, 149, 170,  86,   3,  46,  36, 139,
               128,  68,   2, 101,  90,  43, 106,  91, 181,  11, 194, 111,  79,
               123, 140,  81, 122,  46,  57, 160, 109,  46,  66, 133,  50,  25,
                79, 119,  23, 115,  88,  15, 102, 107,  95,   7, 108,  35, 112,
               103, 110, 130, 160, 155,  61,  92,  76, 102, 160,  91,  18, 168,
               145, 128,   1,  19,  16,  73,  32,  51,  36,  96, 198, 102, 157,
               149, 154,  58,  45, 124, 146,  62, 165, 193,  56, 193, 149,  20,
               180, 161,  57, 183,  62,  80,  61,  67, 148,  98, 150, 163,  61,
                56, 114,  22, 155, 149,  39,  14,  11,  93,   6,   6, 162, 116,
               115,  41,  73, 195,  72, 101,  67,  11, 154,  47,  78,   9, 103,
               197,  19, 162,  70, 136, 186, 140, 167, 115,  68,  46,  71,  73,
                89,  86, 112, 177, 111,  67,  28,  13, 189, 118, 189, 107,  75,
               128, 140, 131,  42, 145, 172, 132,  82,  48,  21,  26, 132, 100,
                57, 165,  85, 135,  80,  54, 125, 128, 110, 104,  79, 184, 173,
                27, 131, 108, 184,  13, 125], dtype=int32)
```

```
In [ ]: # arr filter all those items that are less than equal to 100
        # count =0
        # for item in arr4 :
        #   if (item<100 or item == 100) :
        #     count+=1
        # print(count)

        # or we can do that

        arr4<=100  # give true and false in array
        arr4[arr4<=100]  #filter above arr4 which have true
        arr4[arr4<=100].size
```

```
Out[ ]: 246
```

```
In [ ]: len(arr4[arr4<=100])
```

```
Out[ ]: 246
```

creating 2 D random array

```
In [ ]: arr2 = np.random.randint(1,200,(10,6))
        arr2
```

```
Out[ ]: array([[ 62, 117,  81, 118,  99, 106],
               [ 57, 190,  46, 194,  76,  80],
               [ 89, 180, 123,   8,  26, 123],
               [134,  57, 193, 172, 116, 154],
               [190, 198, 116,  51,  21, 185],
               [138, 143, 116, 124,  42, 190],
               [172,  31,  67, 191, 145,  71],
               [ 95, 193, 120, 105,  94,  51],
               [  9, 195, 104, 145,  18,  81],
               [158, 194,  25, 144, 105, 141]], dtype=int32)
```

creating random 3 d array

```
In [ ]: arr2 = np.random.randint(1,200,(10,6,3))
        arr2
```

```
Out[ ]:  array([[[176,  34,  81],
                [118, 148, 177],
                [  9, 145,  66],
                [ 81, 111, 100],
                [158,  74,  98],
                [ 15,   9, 187]],

               [[ 52,  34,   2],
                [ 15, 190,   3],
                [ 83, 146,  92],
                [ 90,  51, 190],
                [ 35,  60,  51],
                [ 64, 127,  27]],

               [[129,  90,   6],
                [192, 173, 139],
                [116,  65, 176],
                [177, 195,  18],
                [ 97, 127, 170],
                [150, 186, 142]],

               [[154,  60, 182],
                [125, 184, 167],
                [110, 199, 126],
                [114, 118, 197],
                [ 13, 180,  12],
                [142,  37,  89]],

               [[108,  60,   6],
                [  3,  33, 150],
                [ 77,  60,  55],
                [161, 184, 110],
                [164, 150,  96],
                [166, 193,  19]],

               [[ 75, 134,  53],
                [ 59,   7, 150],
                [  6, 123,  10],
                [129, 127, 122],
                [194, 139, 155],
                [ 71, 157,  68]],

               [[ 98, 105, 165],
                [  2, 188, 170],
                [114, 101,  35],
                [ 63, 113,  46],
                [ 50,  85,  77],
                [189,  89,  16]],

               [[124, 150,  22],
                [162, 173,  33],
                [  5,  86, 102],
                [ 80,  19, 143],
                [190,  55,  90],
                [ 86, 122, 158]],

               [[182, 150, 132],
                [194,  71,  97],
                [ 88, 156,  60],
                [126, 147, 118],
                [170, 183,  91],
                [ 80,  29, 184]],

               [[133,  26,  92],
                [ 55,  69, 102],
                [ 80,  31, 112],
                [142,  69,  59],
                [172,  31, 125],
                [199,  24,  89]]], dtype=int32)
```

In [ ]:
```
arr = np.zeros(10)  #single dimensional array with 10 element having 0 value
arr
```

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

In [ ]:
```
arr = np.zeros((10,5))   #2 d array
arr
```

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

In [ ]:
```
arr = np.ones(10)
arr1 = np.ones((10,5))
```

In [ ]:
```
ls = list(range(0,10))   #creating a list in a range
ls
```

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

In [ ]:
```
# arange create only single dimension array
arr = np.arange(60)  #creating an array in a range
arr
```

Out[ ]:
```
array([ 0,  1,  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, 56, 57, 58, 59])
```

If we want to convert 1D into 2D array then we have to decide no. of rows and column

In [ ]:
```
arr = arr.reshape(10,6)
# arr.reshape(10,7)  # can't divide 60 element into 10,7
arr
```

```
Out[ ]: array([[ 0,  1,  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, 56, 57, 58, 59]])
```

DAY 2 OF NUMPY

Creation of 3D array

```
In [ ]: # Creation of 3D array
        # arr = [2D,2D,2D,2D....]
        # these are lists given below,not actual array
        arr1D = [1,2,3]
        arr2D = [[1,2,3],[1,2,3],[1,2,3]]
        arr3D = [[[1,2,3],[1,2,3],[1,2,3]],[[1,2,3],[1,2,3],[1,2,3]]]
```

```
In [ ]: arr = np.array(arr3D)
```

```
In [ ]: arr3 = np.random.randint(1,200,(3,5,3))
        arr3
        # (table no. =3 , rows no.=5 , column no.=3)
```

```
Out[ ]: array([[[170, 156, 141],
                [  2, 151, 124],
                [110, 107, 172],
                [170,  84,  22],
                [ 60,  11,  94]],

               [[  5,  28,  64],
                [170, 116,  93],
                [ 61,  99, 164],
                [133,  69,  66],
                [ 64, 188, 195]],

               [[ 55, 102,  56],
                [  1, 136,  89],
                [145,  80, 165],
                [117,  91, 192],
                [ 82,  28,  28]]], dtype=int32)
```

```
In [ ]: arr3.shape
```

```
Out[ ]: (3, 5, 3)
```

Indexing of 3 dimension array :- indexing --> arr[table,row,column], arr[table-start:end,row-start:end,column-start:end]

```
In [ ]: arr3[1,4,0:]
        arr3[2,:2,1:]
```

```
Out[ ]:  array([[102,  56],
                [136,  89]], dtype=int32)
```

```
In [ ]:  arr3[:,1,:]  #accessing first row of every table
```

```
Out[ ]:  array([[  2, 151, 124],
                [170, 116,  93],
                [  1, 136,  89]], dtype=int32)
```

Some functionalaty of numpy

```
In [ ]:  arr = np.array([9,7,8])
```

```
In [ ]:  # minimum of array
         min(arr)

         # maximum of array
         max(arr)
```

```
Out[ ]:  np.int64(9)
```

```
In [ ]:  # mean of array
         np.mean(arr)

         # sum of array
         np.sum(arr)
```

```
Out[ ]:  np.int64(24)
```

```
In [ ]:  # index of minimum element
         np.argmin(arr)

         # index of maximum element
         np.argmax(arr)
```

```
Out[ ]:  np.int64(0)
```

```
In [ ]:  # sorting of array
         arr.sort()   #ascending order
         arr
```

```
Out[ ]:  array([7, 8, 9])
```

```
In [ ]:  arr[::-1]  #descending order
```

```
Out[ ]:  array([9, 8, 7])
```

So till now we learn accessing , manipulating an 1d,2d,3d array. creating instant array and sum functions like min,max,sum,mean,sort,argmin,argmax etc.