NUMPY

NUMPY is a package for python programming language and simplify operations on multi-dimensional arrays. **NUMPY** was introduced(by Travis Oliphant) in 2005 and distributed as open source library. **NUMPY** is not the part of default python and we need to download it by using pip utility.

pip install numpy or conda install numpy

Diff bw list and numpy array:

list	Numpy array
any type	similer type
of values	of values
resizable	immutable in size
less function	huge library
does not broadcasting	supports broadcasting
does not store actual elements . it store referance only	store actual elements

In [34]:
```
import numpy as np
```

In [35]:
```
salary = np.loadtxt('sal2.txt', delimiter=',')
salary
```

Out[35]:
```
array([5000., 4000., 6000., 3000., 3500., 6000., 8000., 6500.])
```

ndim() function give dimension on numpy array

In [36]:
```
np.ndim(salary)
```

Out[36]:
```
1
```

In [37]:
```
n1 = np.array([[1,2,3], [1,2]])
n1
```

Out[37]:
```
array([list([1, 2, 3]), list([1, 2])], dtype=object)
```

In [38]:
```
np.ndim(n1)
```

Out[38]:
```
1
```

make array arr using list l1

In [39]:
```
l1 = [[1,2,3],[4,5,6], [7,8,9]]
print("l1 = ",l1)
arr = np.array(l1)
arr
```

```
l1 =  [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

Out[39]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

How to create Numpy Array:

1. a=np.array(seq)
2. a=np.arange(start,stop,step)
3. a=np.zeros(size)
4. a=np.ones(size)
5. a=np.empty(size) #generate random float or based on type
6. a=np.linspace(low,high,num)
7. a=np.full(size,initial_value)
8. a=np.random.random(size) #generate random float
9. a=np.random.randint(low,high,size)
10. a=np.random.uniform(low,high,size)

Note:each numpy array is represented by an object of ndarray class.

In [40]:
```
a1=np.array([1,2,3,4])
a2=np.array([[1,2],[3,4]])
print(a1.ndim,type(a1))
print(a2.ndim,type(a2))
print(len(a1),a1.size)
print(len(a2),a2.size)
print(a1)
print(a2)
```

```
1 <class 'numpy.ndarray'>
2 <class 'numpy.ndarray'>
4 4
2 4
[1 2 3 4]
[[1 2]
 [3 4]]
```

Type of array

In [41]:
```
l=[1,2.5,3.1,4]
a=np.array(l,dtype=np.int16)
print(l)
print(a)
```

```
[1, 2.5, 3.1, 4]
[1 2 3 4]
```

Numpy aaray store only similar type of values

In [42]:
```
l=[1,2.5,3.1,4,'hi','india']
a=np.array(l)
print(l)
print(a)
```

```
[1, 2.5, 3.1, 4, 'hi', 'india']
['1' '2.5' '3.1' '4' 'hi' 'india']
```

In [43]:
```
l=[1,2.5,3.1,4,'hi','india']
a=np.array(l,dtype=np.int16)
print(l)
print(a)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-43-b7264e5eaea4> in <module>
      1 l=[1,2.5,3.1,4,'hi','india']
----> 2 a=np.array(l,dtype=np.int16)
      3 print(l)
      4 print(a)

ValueError: invalid literal for int() with base 10: 'hi'
```

In [44]:
```
l=[10,12,-10,5,20]
a=np.array(l,dtype=np.uint16)
print(l)
print(a)
```

```
[10, 12, -10, 5, 20]
[   10    12 65526     5    20]
```

way of creating numpy aaray

In [45]:
```
a=np.arange(1,2,.1)
print(a)
```

```
[1.  1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9]
```

In [48]:
```
a=np.zeros(5,dtype=np.int)
a=np.zeros(5)
print(a)
```

```
[0. 0. 0. 0. 0.]
```

In [49]:
```
a=np.ones(5)
print(a)
```

```
[1. 1. 1. 1. 1.]
```

In [50]:
```
a=np.full(5,5)
print(a)
```

```
[5 5 5 5 5]
```

In [51]:
```
a=np.full(5,6)
print(a)
```

```
[6 6 6 6 6]
```

In [52]:
```
a=np.linspace(1,200,num=2)
a
```

Out[52]:
```
array([  1., 200.])
```

In [53]:
```
a=np.linspace(1,200)#default value of num=50
a
```

Out[53]:
```
array([  1.        ,   5.06122449,   9.12244898,  13.18367347,
        17.24489796,  21.30612245,  25.36734694,  29.42857143,
        33.48979592,  37.55102041,  41.6122449 ,  45.67346939,
        49.73469388,  53.79591837,  57.85714286,  61.91836735,
        65.97959184,  70.04081633,  74.10204082,  78.16326531,
        82.2244898 ,  86.28571429,  90.34693878,  94.40816327,
        98.46938776, 102.53061224, 106.59183673, 110.65306122,
       114.71428571, 118.7755102 , 122.83673469, 126.89795918,
       130.95918367, 135.02040816, 139.08163265, 143.14285714,
       147.20408163, 151.26530612, 155.32653061, 159.3877551 ,
       163.44897959, 167.51020408, 171.57142857, 175.63265306,
       179.69387755, 183.75510204, 187.81632653, 191.87755102,
       195.93877551, 200.        ])
```

In [54]:
```
a=np.linspace(1,200,num=10,dtype=np.int)
a
```

Out[54]:
```
array([  1,  23,  45,  67,  89, 111, 133, 155, 177, 200])
```

create garbage array empty function

In [55]:
```
a=np.empty(2,dtype=np.int)
print(a)
```

```
[4607182418800017408 4641240890982006784]
```

In [56]:
```
np.random.random(5)
```

Out[56]:
```
array([0.34904576, 0.36708857, 0.19696738, 0.69853462, 0.86893617])
```

In [57]:
```
np.random.randint(15,20,5)
```

Out[57]:
```
array([16, 15, 15, 15, 16])
```

In [58]:
```
np.random.uniform(1,5,5)
```

Out[58]:
```
array([3.4505837 , 2.58629694, 4.23357085, 1.16556692, 4.5707419 ])
```

Standard Deviation of aaray a

In [59]:
```
a
```

Out[59]:
```
array([4607182418800017408, 4641240890982006784])
```

In [60]:
```
np.std(a)
```

Out[60]:
```
1.7029236090994688e+16
```

mean of aaray a

In [61]:
```
a.mean()
```

Out[61]:
```
4.624211654891012e+18
```

In [62]:
```
np.mean(a)
```

Out[62]:
```
4.624211654891012e+18
```

In [63]:
```
a
```

Out[63]:
```
array([4607182418800017408, 4641240890982006784])
```

Median of aaray a

In [64]:
```
np.median(a)
```

Out[64]:
```
4.624211654891012e+18
```

In [65]:
```
a.median()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-65-d799d54f1221> in <module>
----> 1 a.median()

AttributeError: 'numpy.ndarray' object has no attribute 'median'
```

In [66]:
```
np.mod(a, 10)
```

Out[66]:
```
array([8, 4])
```

In [67]:
```
a
```

Out[67]:
```
array([4607182418800017408, 4641240890982006784])
```

In [68]:
```
a = np.arange(100,10)
a
```

Out[68]:
```
array([], dtype=int64)
```

In [69]:
```
a = np.arange(9,10)
a
```

Out[69]:
```
array([9])
```

In [70]:
```
a = np.arange(100,1000, 100)
a
```

Out[70]:
```
array([100, 200, 300, 400, 500, 600, 700, 800, 900])
```

In [71]:
```
a.max()
```

Out[71]:
```
900
```

In [72]:
```
np.max(a)
```

Out[72]:
```
900
```

In [73]:
```
np.min(a)
```

Out[73]:
```
100
```

In [74]:
```
a.min()
```

Out[74]:
```
100
```

return index of max or min element of given array np.argmax() and np.argmin()

In [75]:
```
print(np.argmax(a))
print(np.argmin(a))
```

```
8
0
```

In [76]:
```
a = np.random.randint(10, 20, 10)
a
```

Out[76]:
```
array([19, 19, 18, 18, 18, 19, 14, 10, 19, 12])
```

In [77]:
```
np.sort(a)
```

Out[77]:
```
array([10, 12, 14, 18, 18, 18, 19, 19, 19, 19])
```

In [78]:
```
a.sort()
a
```

Out[78]:
```
array([10, 12, 14, 18, 18, 18, 19, 19, 19, 19])
```

In [79]:
```
np.square(a)
```

Out[79]:
```
array([100, 144, 196, 324, 324, 324, 361, 361, 361, 361])
```

In [80]:
```
a.square()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-80-32aa3ac58ac6> in <module>
----> 1 a.square()

AttributeError: 'numpy.ndarray' object has no attribute 'square'
```

In [81]:
```
b = np.square(a)
b
```

Out[81]:
```
array([100, 144, 196, 324, 324, 324, 361, 361, 361, 361])
```

In [82]:
```
b.sqrt()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-82-0da6feebad6f> in <module>
----> 1 b.sqrt()

AttributeError: 'numpy.ndarray' object has no attribute 'sqrt'
```

In [83]:
```
np.sqrt(b)
```

Out[83]:
```
array([10., 12., 14., 18., 18., 18., 19., 19., 19., 19.])
```

In [84]:
```
a.log()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-84-90cf47780a7e> in <module>
----> 1 a.log()

AttributeError: 'numpy.ndarray' object has no attribute 'log'
```

In [85]:
```
np.log(a)
```

Out[85]:
```
array([2.30258509, 2.48490665, 2.63905733, 2.89037176, 2.89037176,
       2.89037176, 2.94443898, 2.94443898, 2.94443898, 2.94443898])
```

In [86]:
```
np.log10(a)
```

Out[86]:
```
array([1.        , 1.07918125, 1.14612804, 1.25527251, 1.25527251,
       1.25527251, 1.2787536 , 1.2787536 , 1.2787536 , 1.2787536 ])
```

In [87]:
```
np.exp(a)
```

Out[87]:
```
array([2.20264658e+04, 1.62754791e+05, 1.20260428e+06, 6.56599691e+07,
       6.56599691e+07, 6.56599691e+07, 1.78482301e+08, 1.78482301e+08,
       1.78482301e+08, 1.78482301e+08])
```

In [88]:
```
a.exp()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-88-95e1b9179fca> in <module>
----> 1 a.exp()

AttributeError: 'numpy.ndarray' object has no attribute 'exp'
```

In [89]:
```
a
```

Out[89]:
```
array([10, 12, 14, 18, 18, 18, 19, 19, 19, 19])
```

In [90]:
```
np.unique(a)
```

Out[90]:
```
array([10, 12, 14, 18, 19])
```

In [91]:
```
binarr = [1,0,0,1,0,1,1,1,0]
np.bincount(binarr)
```

Out[91]:
```
array([4, 5])
```

Bincount function count start from zero to max number of given array

In [92]:
```
a=np.array([1,2,3,2,1,3,4,1,5,6,1,5])
print(np.bincount(a))
```

```
[0 4 2 2 1 2 1]
```

In [93]:
```
np.insert(a,1,100)
```

Out[93]:
```
array([  1, 100,   2,   3,   2,   1,   3,   4,   1,   5,   6,   1,   5])
```

In [94]:
```
np.insert(a,1,[100,200,300])
```

Out[94]:
```
array([  1, 100, 200, 300,   2,   3,   2,   1,   3,   4,   1,   5,   6,
         1,   5])
```

In [95]:
```
np.insert(a,[1,3,6],[100,200,300])
```

Out[95]:
```
array([  1, 100,   2,   3, 200,   2,   1,   3, 300,   4,   1,   5,   6,
         1,   5])
```

In [96]:
```
a
```

Out[96]:
```
array([1, 2, 3, 2, 1, 3, 4, 1, 5, 6, 1, 5])
```

In [97]:
```
np.delete(a,0)
```

Out[97]:
```
array([2, 3, 2, 1, 3, 4, 1, 5, 6, 1, 5])
```

In [99]:
```
a[0]=60
a
```

Out[99]:
```
array([60,  2,  3,  2,  1,  3,  4,  1,  5,  6,  1,  5])
```

In [101]:
```
np.abs(a)
```

Out[101]:
```
array([60,  2,  3,  2,  1,  3,  4,  1,  5,  6,  1,  5])
```

In [103]:
```
np.absolute(a)
```

Out[103]:
```
array([60,  2,  3,  2,  1,  3,  4,  1,  5,  6,  1,  5])
```

In [105]:
```
np.add(a, 5)
```

Out[105]:
```
array([65,  7,  8,  7,  6,  8,  9,  6, 10, 11,  6, 10])
```

In [104]:
```
np.add(a)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-104-faf0c3f0bf60> in <module>
----> 1 np.add(a)

ValueError: invalid number of arguments
```

In [108]:
```
a
```

Out[108]:
```
array([60,  2,  3,  2,  1,  3,  4,  1,  5,  6,  1,  5])
```

In [109]:
```
np.amax(a)
```

Out[109]:
```
60
```

In [110]:
```
np.amin(a)
```

Out[110]:
```
1
```

In [112]:
```
np.append(a, 70)
```

Out[112]:
```
array([60,  2,  3,  2,  1,  3,  4,  1,  5,  6,  1,  5, 70])
```

In [115]:
```
np.append(a, 80)
```

Out[115]:
```
array([60,  2,  3,  2,  1,  3,  4,  1,  5,  6,  1,  5, 80])
```