```
In [7]: import numpy as np
```

Array creation

```
In [9]: arr=np.array([1,2,3])
        print(arr)
```

```
[1 2 3]
```

```
In [12]: arr=np.array([[1,2,3],[4,5,6],[7,8,9]])
         print(arr)
```

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

```
In [15]: arr1=np.ones(4)
         print(arr1)
```

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

```
In [16]: arr2=np.empty(4)
         print(arr2)
```

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

```
In [14]: arr2=np.ones((3,3))
         print(arr2)
```

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

```
In [17]: arr5=np.concatenate([arr1,arr2])
         print(arr5)
```

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

```
In [18]: arr7=np.zeros_like(arr5)
         print(arr7)
```

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

In [21]:
```
arr9=np.random.random((5,7))
print(arr9)
```

```
[[0.06863913 0.98716719 0.77864514 0.1710649  0.1344111  0.36255731
  0.73584282]
 [0.87939052 0.87887583 0.49871173 0.27208402 0.05207227 0.07549829
  0.02216456]
 [0.35865425 0.42765589 0.77007996 0.46573832 0.12351312 0.3236207
  0.35429239]
 [0.1613377  0.45317323 0.70458256 0.64121805 0.99334486 0.79069979
  0.41407613]
 [0.62223483 0.19089337 0.6789878  0.8277786  0.2030802  0.87789753
  0.16046163]]
```

In [22]:
```
arr11=np.linspace(5,20,5,dtype=float)
print(arr11)
```

```
[ 5.    8.75 12.5  16.25 20.  ]
```

ndarray attributes

In [23]:
```
arr24=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(arr24)
```

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

In [25]:
```
print(arr24.ndim)
```

```
2
```

In [26]:
```
print(arr24.shape)
```

```
(3, 3)
```

In [27]:
```
print(arr.dtype)
```

```
int32
```

In [28]:
```
print(arr.data)
```

```
<memory at 0x0000018BD8C6DD80>
```

In [29]:
```
print(arr.nbytes)
```

```
36
```

ndarray access

```
In [30]: arr=np.array([1,2,3,4,5])
         arr2=np.array([[1,2,3],[4,5,6],[7,8,9]])
         print(arr[3])
```

```
4
```

```
In [31]: print(arr[-3])
```

```
3
```

```
In [32]: print(arr2[1][2])
```

```
6
```

```
In [33]: print(arr2)
```

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

```
In [34]: print(arr)
```

```
[1 2 3 4 5]
```

ndarray reference

```
In [35]: arr=np.array([25,87,36,14])
         print(arr)
```

```
[25 87 36 14]
```

```
In [36]: arr3=arr
         arr3[2]=20
         print(arr3)
```

```
[25 87 20 14]
```

nd array shaping

```
In [39]: arr=np.array([7,8,6,4,8,3,2,5,6,15])
         print(arr)
```

```
[ 7  8  6  4  8  3  2  5  6 15]
```

```
In [41]: print(arr.shape)
```

```
(10,)
```

In [46]:
```
arr1=arr.reshape(5,2)
print(arr1.shape)
print(arr1)
```

```
(5, 2)
[[ 7  8]
 [ 6  4]
 [ 8  3]
 [ 2  5]
 [ 6 15]]
```

In [47]:
```
arr2=arr.flatten()
arr3=arr.ravel()
print(arr2)
```

```
[ 7  8  6  4  8  3  2  5  6 15]
```

In [49]:
```
print(arr3)
```

```
[ 7  8  6  4  8  3  2  5  6 15]
```

ndarray slicing

In [50]:
```
arr=np.array([1,2,3,4,5])
arr2=np.array([[1,2,3,4,5],[6,7,8,9,10]])
print(arr[1:4])
```

```
[2 3 4]
```

In [51]:
```
print(arr[4:])
```

```
[5]
```

In [52]:
```
print(arr[:4])
```

```
[1 2 3 4]
```

In [53]:
```
print(arr[1:5:2])
```

```
[2 4]
```

In [54]:
```
print(arr[::2])
```

```
[1 3 5]
```

In [55]:
```
print(arr2[0:2,1:4])
```

```
[[2 3 4]
 [7 8 9]]
```

Advanced indexing

```
In [56]: arr=np.array([1,2,3,4,5])
         arr1=np.array([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15]])
         list1=[1,2,-3]
         print(arr[list1])
```

```
[2 3 3]
```

```
In [57]: check=arr>3
         print(check)
```

```
[False False False  True  True]
```

```
In [58]: print(arr[check])
```

```
[4 5]
```

```
In [59]: l=[0,1,0,1,0]
         print(arr[l])
```

```
[1 2 1 2 1]
```

Numpy methods

```
In [60]: arr=np.array([1,2,3,4,5])
         arr1=np.array([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15]])
         x=arr1.flatten()
         print(x)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15]
```

```
In [61]: x[4]=30
         y=np.ravel(arr1)
         y[3]=20
         print(arr1); print(x); print(y)
```

```
[[ 1  2  3 20  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]]
[ 1  2  3  4 30  6  7  8  9 10 11 12 13 14 15]
[ 1  2  3 20  5  6  7  8  9 10 11 12 13 14 15]
```

```
In [62]: x.resize((3,5))
         print(arr1)
```

```
[[ 1  2  3 20  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]]
```

In [63]:
```
print(arr1.swapaxes(1,0))
```
```
[[ 1  6 11]
 [ 2  7 12]
 [ 3  8 13]
 [20  9 14]
 [ 5 10 15]]
```

In [64]:
```
print(arr1.dtype)
```
```
int32
```

In [65]:
```
print(arr1.astype(float))
```
```
[[ 1.  2.  3. 20.  5.]
 [ 6.  7.  8.  9. 10.]
 [11. 12. 13. 14. 15.]]
```

In [66]:
```
arr1[1,3]=0;arr1[0,3]=0
print(arr1.nonzero())
```
```
(array([0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 2], dtype=int64), array([0, 1, 2,
4, 0, 1, 2, 4, 0, 1, 2, 3, 4], dtype=int64))
```

In [67]:
```
print(arr1)
```
```
[[ 1  2  3  0  5]
 [ 6  7  8  0 10]
 [11 12 13 14 15]]
```

In [68]:
```
arr1.argsort(axis=0)
print(arr1.argsort(axis=1))
```
```
[[3 0 1 2 4]
 [3 0 1 2 4]
 [0 1 2 3 4]]
```

In [69]:
```
arr1.sort(axis=1)
print(arr1)
```
```
[[ 0  1  2  3  5]
 [ 0  6  7  8 10]
 [11 12 13 14 15]]
```

In [70]:
```
print(arr.searchsorted(3))
```
```
2
```

Iterating arrays

In [7]:
```
import numpy as np
```

In [8]:
```
arr=np.array([1,2,3,4,5])
arr1=np.array([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15]])
for x in arr:
    print(x)
for x in arr1:
    print(x)
for x in arr1:
    for y in x:
        print(y)
for x in np.nditer(arr1):
    print(x)
```

```
1
2
3
4
5
[1 2 3 4 5]
[ 6  7  8  9 10]
[11 12 13 14 15]
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
```

String Functions

In [10]:
```
np.char.add(['a','b'],['c','d'])
```

Out[10]:
```
array(['ac', 'bd'], dtype='<U2')
```

In [11]:
```
np.char.multiply(['a','b'],3)
```

Out[11]:
```
array(['aaa', 'bbb'], dtype='<U3')
```

Array Operations

In [15]:
```
arr1=np.array([[1,2,3],[4,5,6],[7,8,9]])
arr2=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(arr1+arr2)
print(np.add(arr1,arr2))
```

```
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]
```

In [16]:
```
arr1=np.array([[1,2,3],[4,5,6],[7,8,9]])
arr2=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(arr1==arr2)
print(np.equal(arr1,arr2))
```

```
[[ True  True  True]
 [ True  True  True]
 [ True  True  True]]
[[ True  True  True]
 [ True  True  True]
 [ True  True  True]]
```

Array Calculation Methods

In [17]:
```
arr1=np.array([[1,2,3],[4,5,6],[7,8,9]])
arr2=np.array([[-1.7,2.1,3.5],[4,-5,6],[7,8,-9]])
print(arr1.sum())
print(arr1.sum(axis=0))
print(arr1.sum(axis=1))
print(arr1.prod())
```

```
45
[12 15 18]
[ 6 15 24]
362880
```

In [18]:
```
print(np.sin(arr2))
print(np.negative(arr2))
print(np.ceil(arr2))
print(np.minimum(arr1,arr2))
```

```
[[-0.99166481  0.86320937 -0.35078323]
 [-0.7568025   0.95892427 -0.2794155 ]
 [ 0.6569866   0.98935825 -0.41211849]]
[[ 1.7 -2.1 -3.5]
 [-4.   5.  -6. ]
 [-7.  -8.   9. ]]
[[-1.  3.  4.]
 [ 4. -5.  6.]
 [ 7.  8. -9.]]
[[-1.7  2.   3. ]
 [ 4.  -5.   6. ]
 [ 7.   8.  -9. ]]
```

Broadcasting

In [19]:
```
arr1 = np.array([[0],[10],[20],[30]])
arr2 = np.array([0,1,2])
print(arr1+arr2)
```

```
[[ 0  1  2]
 [10 11 12]
 [20 21 22]
 [30 31 32]]
```

Conditional

In [20]:
```
arr1=np.array([[-1.7,2.1,3.5],[4,
-5,6],[7,8,-9]])
print(arr1>0)
```

```
[[False  True  True]
 [ True False  True]
 [ True  True False]]
```

In [21]:
```
print((arr1>2).sum())
print(np.where(arr1>3,'Yes','No'))
```

```
6
[['No' 'No' 'Yes']
 ['Yes' 'No' 'Yes']
 ['Yes' 'Yes' 'No']]
```

Graphical

In [23]:
```
arr1 = np.random.randint(100, size =(50))
np.histogram(arr1, bins = [0, 10, 20, 30, 40,50, 60, 70, 80, 90, 100])
```

Out[23]:
```
(array([6, 2, 6, 8, 3, 7, 2, 3, 5, 8], dtype=int64),
 array([  0,  10,  20,  30,  40,  50,  60,  70,  80,  90, 100]))
```

In [ ]:
```

```