numpy: numerical python

1. Allow several mathematical operations
2. faster operations

In [1]:
```
import numpy as np
```

List Vs Numpy-Time taken

In [2]:
```
from time import process_time
```

time taken by a list

In [3]:
```
python_list=[i  for i in range(10000)]
start_time=process_time()
python_list=(i+5 for i in python_list)
end_time=process_time()
print(end_time-start_time)
```

```
0.015625
```

In [4]:
```
np_array=np.array([i for i in range(10000)])
start_time=process_time()
```

In [5]:
```
np_array+=5 #np_Array=np_array=5

end_time=process_time()
print(end_time-start_time)
```

```
0.015625
```

numpy array

In [6]:
```
#list
list1=[1,2,3,4,5]
print(list1)
type(list1)
```

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

Out[6]:
```
list
```

In [10]:
```
import numpy as np
np =np.array([1, 2, 3, 4, 5])
print(np[0])
print(np[1])
```

```
1
2
```

In [11]:
```
import numpy as np
list2 = np.array([1, 2, 3, 4, 5])
print(list2[2])
```

```
3
```

In [13]:
```
b=np.array([(1,2,3),(4,5,6)])
print(b[0])
print(b[1])
```

```
[1 2 3]
[4 5 6]
```

In [14]:
```
b.shape
```

Out[14]:
```
(2, 3)
```

In [15]:
```
c=np.array([(1,2,3,4),(5,6,7,8)],dtype=float)
```

In [16]:
```
c
```

Out[16]:
```
array([[1., 2., 3., 4.],
       [5., 6., 7., 8.]])
```

initial placeholders in numpy arrays

In [18]:
```
#created a numpy array of zeroes
x=np.zeros((4,5))
print(x)
```

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

In [19]:
```
#creates a numpy array of ones
y=np.ones((3,3))
print(y)
```

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

In [21]:
```
#array of a particular value
z=np.full((5,4),5)
print(z)
```

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

```
In [22]: #create an identity matrix
         x=np.eye(4)
         x

Out[22]: array([[1., 0., 0., 0.],
                [0., 1., 0., 0.],
                [0., 0., 1., 0.],
                [0., 0., 0., 1.]])

In [23]: #create a numpy array with random values
         b=np.random.random((3,4))

In [24]: b

Out[24]: array([[0.12969877, 0.82657898, 0.07648113, 0.11767547],
                [0.95264415, 0.98621162, 0.71389104, 0.21557665],
                [0.35184955, 0.22024712, 0.05690899, 0.94399145]])

In [26]: #random integer  values  arrays within  a specific range
         c=np.random.randint(10,100,(3,5))
         c

Out[26]: array([[35, 93, 84, 68, 67],
                [52, 35, 22, 29, 22],
                [21, 75, 78, 37, 42]])

In [28]: # array of evenly spaced values---->specifying the number of values requaire
         d=np.linspace(10,30,5)
         print(d)

         [10. 15. 20. 25. 30.]

In [29]: #array of evenly spaced values----->specifying the step
         e=np.arange(10,30,5)
         print(e)

         [10 15 20 25]

In [32]: #convert a list to a numpy array
         list2=[10,20,20,20,50]
         np_array=np.asarray(list2)
         print(np_array)
         type(np_array)

         [10 20 20 20 50]

Out[32]: numpy.ndarray
```

Analysing a numpy array

In [33]:
```
c=np.random.randint(10,50,(5,5))
print(c)
```

```
[[14 32 26 37 13]
 [46 31 12 35 33]
 [13 27 30 31 15]
 [31 37 39 46 14]
 [34 34 30 44 28]]
```

In [34]:
```
#array dimension
print(c.shape)
```

```
(5, 5)
```

In [35]:
```
#number of dimension
print(c.ndim)
```

```
2
```

In [36]:
```
#the number of element in an array
print(c.size)
```

```
25
```

In [37]:
```

#checking the data type of the values in the array
print(c.dtype)
```

```
int32
```

mathmaitical operation on numpy

In [38]:
```
list1=[1,2,3,4,5]
list2=[6,7,8,9,10]
print(list1+list2)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

In [40]:
```
a=np.random.randint(0,10,(3,3))
b=np.random.randint(10,20,(3,3))
print(a)
print(b)
```

```
[[1 2 2]
 [2 7 0]
 [7 8 2]]
[[19 17 10]
 [19 19 10]
 [11 17 12]]
```

In [41]:
```
print(a+b)
```

```
[[20 19 12]
 [21 26 10]
 [18 25 14]]
```

In [42]:
```
print(a-b)
```
```
[[-18 -15  -8]
 [-17 -12 -10]
 [ -4  -9 -10]]
```

In [43]:
```
print(a%b)
```
```
[[1 2 2]
 [2 7 0]
 [7 8 2]]
```

In [44]:
```
print(a/b)
```
```
[[0.05263158 0.11764706 0.2       ]
 [0.10526316 0.36842105 0.        ]
 [0.63636364 0.47058824 0.16666667]]
```

In [45]:
```
print(a*b)
```
```
[[ 19  34  20]
 [ 38 133   0]
 [ 77 136  24]]
```

In [46]:
```
print(np.add(a,b))
```
```
[[20 19 12]
 [21 26 10]
 [18 25 14]]
```

In [48]:
```
a.shape
```
Out[48]:
```
(3, 3)
```

In [50]:
```
#transpose
trans=np.transpose(b)
```

In [51]:
```
trans
```
Out[51]:
```
array([[19, 19, 11],
       [17, 19, 17],
       [10, 10, 12]])
```

In [52]:
```
trans.shape
```
Out[52]:
```
(3, 3)
```

In [54]:
```
#reshaping a array
a=np.random.randint(0,10,(2,3))
```

In [55]:
```
a
```
Out[55]:
```
array([[8, 9, 0],
       [8, 8, 5]])
```

In [56]:
```
b=a.reshape(3,2)
print(b.shape)
```

```
(3, 2)
```

In [ ]:
```
Thanku
```