Numpy

Numpy is library which is used for the scientific computing in python. It provides multidimensional array object. Compare to list numpy is far better in all aspects like performance, memory efficiency, operations, Usecase etc.

Numpy is built by- mathematical + Data Structures + statistics + linear albgebra.

Numpy is the library which handles the array. In numpy we have 1d array, 2d array, 3d array, nd array

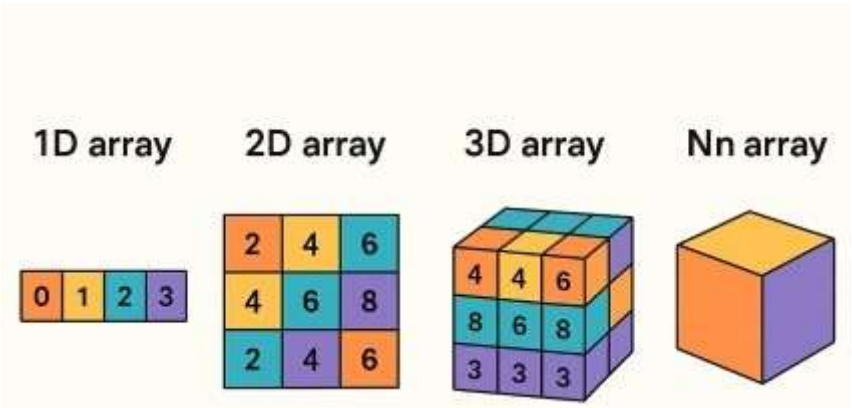
Array==Tables==rows*columns

1D array as a simple row of values

2D array as a matrix (rows × columns)

3D array as a stack of matrices (like layers)

ND array abstractly as a generalized cube or labeled n-dimensional block



Numpy uses in : Arithematic operations, Statistical operations, bitwise operation, sorting & counting, mathematical operations, Brodcasting, Linear algebra, Matrix operation.

Data analyst and data scientist will word on nd array. Numpy is very powerful libraray for bussiness analyst, data analyst, python developer, data scientists. Data analyst and data scientists doesn't required dsa because libraraies and packages will take care.

Basically system wont understand the image directly so we convert image to vector form. Numpy is use to convert and it is very powerful.

Numpy functions like :

1. importing numpy
2. creating arrays
3. array operations
4. Array manipulation
5. statistical operations
6. indexing and slicing
7. logical operations
8. broadcasting
9. concatenation
10. stacking
11. linear algebra
12. random sampling
13. avoiding copy
14. Handling nan
15. vectorized operations
16. save and load
17. memeory usage

Numpy :

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

In [8]:
```
np.__version__ # to know the version
```

Out[8]:
```
'1.26.4'
```

Range

It is a built-in function used to generate a sequence of numbers in python

In [10]:
```
range(5)
```

Out[10]:
```
range(0, 5)
```

In [11]:
```
list(range(5))
```

Out[11]:
```
[0, 1, 2, 3, 4]
```

In [12]:
```
r= range(10,30,5) # we can pass 3 aruguments in range function
r
```

Out[12]:
```
range(10, 30, 5)
```

In [13]:
```
for i in r:
    print(i)
```

```
10
15
20
25
```

In [14]:
```
r1=range(10,20)
r1
```

Out[14]:
```
range(10, 20)
```

In [15]:
```
for i in r1:
    print(i)
```

```
10
11
12
13
14
15
16
17
18
19
```

Creating Array

In [17]:
```
a=np.array([3,5,64,22,45,78,2,1])
a
```

Out[17]:
```
array([ 3,  5, 64, 22, 45, 78,  2,  1])
```

In [18]:
```
type(a)
```

Out[18]:
```
numpy.ndarray
```

converting list to array

In [20]:
```
l=[2,3,5,6,22,45,1]
l
```

Out[20]:
```
[2, 3, 5, 6, 22, 45, 1]
```

In [21]:
```
type(l)
```

Out[21]:
```
list
```

In [22]:
```
a=np.array(l)
a
```

Out[22]:
```
array([ 2,  3,  5,  6, 22, 45,  1])
```

In [23]:
```
type(a)
```

Out[23]:
```
numpy.ndarray
```

arange-

arange() : is a NumPy function used to create arrays with regularly spaced values

In [25]:
```
np.arange(10)    # 1d array
```

Out[25]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [26]:
```
np.arange(5.0)
```

Out[26]:
```
array([0., 1., 2., 3., 4.])
```

In [27]:
```
np.arange(10,30,5)  # prints range from 10,30 with step of 5
```

Out[27]:
```
array([10, 15, 20, 25])
```

In [28]:
```
np.arange(4,12)
```

Out[28]:
```
array([ 4,  5,  6,  7,  8,  9, 10, 11])
```

In [29]:
```
np.arange(30,20)  # the 1st arug < 2nd arug i.e 20,30
```

Out[29]:
```
array([], dtype=int32)
```

In [30]:
```
np.arange(-20,-7)
```

Out[30]:
```
array([-20, -19, -18, -17, -16, -15, -14, -13, -12, -11, -10,  -9,  -8])
```

In [31]:
```
np.arange(-10,3)
```

Out[31]:
```
array([-10,  -9,  -8,  -7,  -6,  -5,  -4,  -3,  -2,  -1,   0,   1,   2])
```

we will create by using varaiable

In [33]:
```
n=np.arange(5,10)
n
```

Out[33]:
```
array([5, 6, 7, 8, 9])
```

In [34]:
```
n1=np.arange(3,3)
n1
```

Out[34]:
```
array([], dtype=int32)
```

In [35]:
```
np.arange(15,35,3)  # prints range from 15,35 with step of 3
```

Out[35]:
```
array([15, 18, 21, 24, 27, 30, 33])
```

Zeros

Return a new array of given shape and type, filled with zeros.

In [37]:
```
np.zeros(5)  # default it will print float value
```

Out[37]:
```
array([0., 0., 0., 0., 0.])
```

In [38]:
```
np.zeros(5,dtype=int)
```

Out[38]:
```
array([0, 0, 0, 0, 0])
```

In [39]:
```
np.zeros((2,2),dtype=int)
```

Out[39]:
```
array([[0, 0],
       [0, 0]])
```

In [40]:
```
np.zeros((3,2)) # 3 rows and 2 columns
```

Out[40]:
```
array([[0., 0.],
       [0., 0.],
       [0., 0.]])
```

In [41]:
```
np.zeros((10,10))
```

Out[41]:
```
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.],
       [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 [42]:
```
np.zeros((10,10),dtype=int)
```

Out[42]:
```
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],
       [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]])
```

ones

Return a new array of given shape and type, filled with ones.

In [44]:
```
np.ones(5)
```

Out[44]:
```
array([1., 1., 1., 1., 1.])
```

In [45]:
```
np.ones(5,dtype=int)
```

Out[45]:
```
array([1, 1, 1, 1, 1])
```

In [46]:
```
np.ones((2,3))
```

Out[46]:
```
array([[1., 1., 1.],
       [1., 1., 1.]])
```

In [47]:
```
np.ones((2,3),dtype=int)
```

Out[47]:
```
array([[1, 1, 1],
       [1, 1, 1]])
```

In [48]:
```
np.ones((10,10))
```

Out[48]:
```
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., 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., 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 [49]:
```
np.twos(4)  # numpy contain only ones and zeros
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[49], line 1
----> 1 np.twos(4)

File ~\anaconda3\Lib\site-packages\numpy\__init__.py:333, in __getattr__(attr)
    330     "Removed in NumPy 1.25.0"
    331     raise RuntimeError("Tester was removed in NumPy 1.25.")
--> 333 raise AttributeError("module {!r} has no attribute "
    334                      "{!r}".format(__name__, attr))

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

random.rand

Random values in a given shape

np=package

random=module

rand=function

In [50]:
```
np.random.rand(5)
```

Out[50]:
```
array([0.90861798, 0.64941397, 0.35662788, 0.4367931 , 0.573868  ])
```

In [76]:
```
np.random.rand(20) # it prints 20 random numbers
```

Out[76]:
```
array([0.58292518, 0.32804785, 0.12993141, 0.68253386, 0.8458852 ,
       0.32221489, 0.23824275, 0.95167274, 0.12385408, 0.3357349 ,
       0.98409804, 0.4400645 , 0.69681318, 0.41793323, 0.13189425,
       0.61683624, 0.61971729, 0.83480476, 0.73026599, 0.34607934])
```

In [82]:
```
np.random.randint(20) # here we defined the type like int so it genertes 1 number b
```

Out[82]:
```
7
```

In [52]:
```
np.random.rand(2,3) # 2 rows 3, columns
```

Out[52]:
```
array([[0.46410291, 0.25464146, 0.14466215],
       [0.99272082, 0.57792004, 0.60318966]])
```

In [54]:
```
np.random.rand(3,3) # 3 rows and 3 columns
```

Out[54]:
```
array([[0.186601  , 0.0639722 , 0.77988129],
       [0.29218014, 0.55719187, 0.96299003],
       [0.0191287 , 0.62349429, 0.20036091]])
```

In [60]:
```
np.random.randint(2,10)  # it will give the random number
```

Out[60]:
```
7
```

In [64]:
```
np.random.randint(2,15,10) # 10 random number b/w 2 to 25
```

Out[64]:
```
array([10,  5,  3, 10, 14, 10, 10, 12, 10, 13])
```

In [68]:
```
np.random.randint(1,50,(5,5)) # 5*5 matrix it will print in b/w 1 to 50
```

Out[68]:
```
array([[45,  6, 28, 48, 47],
       [20, 12, 31, 43, 30],
       [ 9,  2, 28, 17, 49],
       [ 5, 45, 25, 34, 46],
       [47,  8, 21,  4, 39]])
```

In [74]:
```
np.random.randint(1,99,(10,10)) # 10*10 matrix it will print in b/w 1 to 99
```

Out[74]:
```
array([[87, 60, 33, 30, 88, 94, 70, 91, 49, 86],
       [17, 82, 28, 15, 36, 28, 84, 95, 83, 33],
       [56, 31, 16, 75,  8, 54, 70, 61, 13, 30],
       [ 4, 57,  8,  8, 18, 43, 32, 50, 77, 42],
       [87, 66, 89, 30, 77, 36, 42, 84,  9, 95],
       [ 3, 16, 14, 64, 84, 19, 97, 88, 79, 31],
       [12, 48, 44, 98, 97, 24, 81, 16, 84, 14],
       [43, 78, 65, 69, 30,  7, 46, 95, 48, 80],
       [35, 10, 95, 27, 77,  1, 92, 14, 43, 35],
       [93, 77, 64, 16, 26, 50, 94, 39, 45, 16]])
```

In [ ]:
```

```