Numpy:

- It is one of the data science library.
- Numpy abbrevation as numerical python.
- It is mainy used for N-dimensional arrays.
- first we need import numpy

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

In [2]:
```
# To know current numpy .
np.__version__
```

Out[2]: '1.20.1'

syntax of numpy array creation:

- variable_name = np.array(list or tuple)

In [5]:
```
# numpy 1-d array creation..
n1 = np.array([1,2,3,4,5])
print(n1)
print(type(n1))
```

```
[1 2 3 4 5]
<class 'numpy.ndarray'>
```

In [11]:
```
# attributes of numpy array..
print(n1)
print(n1.dtype)# display the data type of numpy array
print(n1.size)# display the number of elements
print(n1.itemsize)# display the each item size in bytes.
print(n1.shape)# display the shape
print(n1.ndim)# display the dimensions
```

```
[1 2 3 4 5]
int32
5
4
(5,)
1
```

In [15]:
```
# creation of numpy 2-d array.
n2 = np.array([[1,2,3],[4,5,6]])
print(n2)
print(n2.ndim)
print(n2.shape)
print(n2.size)
```

```
[[1 2 3]
 [4 5 6]]
2
(2, 3)
6
```

In [16]:
```
# To generate the sequnce of numbers from to n using range function?
n3 = np.array(range(1,11))
print(n3)
```

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

In [17]:
```
np.arange(1,14)
```

Out[17]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13])
```

In [18]:
```
np.arange(1.0,10.0)
```

Out[18]:
```
array([1., 2., 3., 4., 5., 6., 7., 8., 9.])
```

In [19]:
```
np.array(range(1.0,10.0))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-19-1c9320a7a8c8> in <module>
----> 1 np.array(range(1.0,10.0))

TypeError: 'float' object cannot be interpreted as an integer
```

- Note

```
- range function -> only accept the integer values
- arange function ->It can accept intergers as well as floatings..
```

In [22]:
```
print(np.array(range(1,10,2)))
print(np.array(range(1,10,3)))
print(np.arange(10,1,-1))
```

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

Array Initialization

In [25]:
```
a1 = np.zeros(2,dtype = int)
print(a1)
print(a1.ndim)
```

```
[0 0]
1
```

In [28]:
```
a2 = np.zeros((2,3),dtype=int)
print(a2)
print(a2.ndim)
```

```
[[0 0 0]
 [0 0 0]]
2
```

In [29]:
```
a3 = np.ones((3,3),dtype=int)# 2-d array
print(a3)
```

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

In [30]:
```
np.ones(4,dtype=int) # 1-d array
```

Out[30]:
```
array([1, 1, 1, 1])
```

In [31]:
```
# To print the identity matrix..
np.eye((3),dtype=int)
```

Out[31]:
```
array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])
```

In [35]:
```
#To print diagonal values..
np.diag([3,4,5])
```

Out[35]:
```
array([[3, 0, 0],
       [0, 4, 0],
       [0, 0, 5]])
```

In [36]:
```
# To print the fill the specific values..
np.full(4,7)# 1-d array
#4 is number elements
# 7 is filling the value
```

Out[36]:
```
array([7, 7, 7, 7])
```

In [38]:
```
np.full((3,4),8)
# here 3 is number of rows
# 4 is number of columns
# 8 is filling value..
```

Out[38]:
```
array([[8, 8, 8, 8],
       [8, 8, 8, 8],
       [8, 8, 8, 8]])
```

In [41]:
```
# To perform the airthemetic operations..
n1 = np.array([[1,2],[3,4]])
n2 = np.array([[1,1],[2,2]])
print(n1)
print("===============")
print(n2)
print("Addition of two matrix..")
print(n1+n2)
print("subtraction of two matrix..")
print(n1-n2)
print("multiplication of two matrix..")
print(n1*n2)
print('division of two matrix...')
print(n1/n2)
```

```
[[1 2]
 [3 4]]
===============
[[1 1]
 [2 2]]
Addition of two matrix..
[[2 3]
 [5 6]]
subtraction of two matrix..
[[0 1]
 [1 2]]
multiplication of two matrix..
[[1 2]
 [6 8]]
division of two matrix...
[[1.  2. ]
 [1.5 2. ]]
```

In [42]:
```
print(n1.dot(n2))# matrix multiplication
print(n1*n2)# element wise product..
```

```
[[ 5  5]
 [11 11]]
[[1 2]
 [6 8]]
```

random

In [47]:
```
# TO generate the single random number in between the 1 to 10..
print(np.random.randint(1,10))
```

```
1
```

In [49]:
```
# To generate the random numbers in between 1 to n?
print(np.random.randint(1,10,5))# 5 is random number count
print(np.random.randint(45,105,7))
```

```
[4 1 4 4 2]
[79 79 95 57 76 81 73]
```

In [50]:
```
# To generate the numpy 2d-array using random?
print(np.random.randint(10,40,(2,3)))# 2 is number os rows and
#3 is number of columns
```

```
[[36 35 37]
 [33 20 20]]
```

numpy array indexing and slicing:

In [51]:
```
a1 = np.arange(1,20)
print(a1)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
```

In [55]:
```
print(a1[4])
print(a1[[2,6,9]])
print(a1[1:7])
```

```
5
[ 3  7 10]
[2 3 4 5 6 7]
```

In [59]:
```
a2 = np.array([[1,2],[3,4]])
print(a2)
print(a2[1,0])#1 is row index number 0 is column index number
# 00 01
# 10 11
print(a2[1,1])
```

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

Array manipulations:

- reshape
- resize

In [61]:
```
# By using the reshape we can convert the 1-d array to n-d array..
n1 = np.arange(16).reshape(4,4)
print(n1)
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]
 [12 13 14 15]]
```

In [62]:
```
np.arange(16).reshape(2,8)
```

Out[62]:
```
array([[ 0,  1,  2,  3,  4,  5,  6,  7],
       [ 8,  9, 10, 11, 12, 13, 14, 15]])
```

In [63]:
```
np.arange(10).reshape(2,5)
```

Out[63]:
```
array([[0, 1, 2, 3, 4],
       [5, 6, 7, 8, 9]])
```

In [64]:
```
np.arange(10).reshape(5,-1)# here -1 is column framing
```

Out[64]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7],
       [8, 9]])
```

In [65]:
```
np.arange(8).reshape(4,-1)
```

Out[65]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

resize:

- syntax: np.resize(arrayname,size)

In [67]:
```
# By using resize we can increase or decrease the numpy array size..
a=np.array([[1,2],[3,4]])
print(a)
print('=============')
print(np.resize(a,(2,3)))
```

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

In [70]:
```
b = np.random.randint(1,10,(3,3))
print(b)
print("==========")
print(np.resize(b,(2,2)))
```

```
[[3 5 7]
 [5 5 7]
 [9 8 2]]
==========
[[3 5]
 [7 5]]
```

In [ ]:
```

```