What is NumPy?

NumPy is a Python library used for working with arrays.

It also has functions for working in domain of linear algebra, fourier transform, and matrices.

Why Use NumPy?

In Python we have lists that serve the purpose of arrays, but they are slow to process.

NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

The array object in NumPy is called **ndarray**, it provides a lot of supporting functions that make working with **ndarray** very easy.

Arrays are very frequently used in data science, where speed and resources are very important.

```
import numpy

arr = numpy.array([1, 2, 3, 4, 5])

print(arr)
```

OR

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])

print(arr)
```

OP -> [1 2 3 4 5]

#Checking Version of Numpy:

```
import numpy as np

print(np.__version__)
```

Numpy also support indexing and slicing.

Data Types in NumPy

NumPy has some extra data types, and refer to data types with one character, like i for integers, u for unsigned integers etc.

Below is a list of all data types in NumPy and the characters used to represent them.

- i - integer
- b - boolean
- u - unsigned integer
- f - float
- c - complex float
- m - timedelta
- M - datetime
- O - object
- S - string
- U - unicode string
- V - fixed chunk of memory for other type (void)

NumPy Array Copy and View

#Copy

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])
x = arr.copy()
arr[0] = 42

print(arr)
print(x)
```

O/P ->

[42 2 3 4 5]

[1 2 3 4 5]

#View

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])

x = arr.view()

x[0] = 31

print(arr)

print(x)
```

O/P ->

[31 2 3 4 5]

[31 2 3 4 5]

Check if Array Owns its Data

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
x = arr.copy()
y = arr.view()
print(x.base)
print(y.base)
```

O/P ->

```
None
[1 2 3 4 5]
```

Shape of an Array

```
import numpy as np
arr = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print(arr.shape)
```

O/P ->

```
(2, 4)
```

Numpy Array Reshape

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
newarr = arr.reshape(4, 3)
print(newarr)
```

O/P ->

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

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
newarr = arr.reshape(2, 3, 2)
print(newarr)
```

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

 [[ 7  8]
  [ 9 10]
  [11 12]]]
```

NumPy Array Iterating

```
import numpy as np
arr = np.array([1, 2, 3])
for x in arr:
  print(x)
```

O/P->

1

2

3

#2D

```
import numpy as np
arr = np.array([[1, 2, 3], [4, 5, 6]])
for x in arr:
  for y in x:
    print(y)
```

O/P ->

1

2

3

4

5

6

NumPy Joining Array

```
import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.concatenate((arr1, arr2))
print(arr)
```

O/P->

[1 2 3 4 5 6]

#2D

```
import numpy as np

arr1 = np.array([[1, 2], [3, 4]])

arr2 = np.array([[5, 6], [7, 8]])

arr = np.concatenate((arr1, arr2), axis=1)

print(arr)
```

O/P->

[[1 2 5 6]

 [3 4 7 8]]

NumPy Splitting Array

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6])
newarr = np.array_split(arr, 3)
print(newarr)
```

O/P - >

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

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6])
newarr = np.array_split(arr, 3)
print(newarr[0])
print(newarr[1])
print(newarr[2])
```

O/P->

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

NumPy Searching Arrays

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x = np.where(arr == 4)
print(x)
```

O/P ->

```
(array([3, 5, 6]),)
```

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8])
x = np.where(arr%2 == 0)
print(x)
```

O/P - >

(array([1, 3, 5, 7]),)

```
import numpy as np
arr = np.array([1, 3, 5, 7])
x = np.searchsorted(arr, [2, 4, 6])
print(x)
```

O/P -> [1 2 3]

Numpy Array Sorting

```
import numpy as np
arr = np.array([3, 2, 0, 1])
print(np.sort(arr))
```

O/P ->

[0 1 2 3]

#2D

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

O/P - >

[[2 3 4]

 [0 1 5]]

Filter Array

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

filter_arr = arr % 2 == 0

newarr = arr[filter_arr]

print(filter_arr)

print(newarr)
```

O/P ->

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

[2 4 6]
```