What is NumPy?

NumPy is a Python library.

NumPy is used for working with arrays.

NumPy is short for "Numerical Python"

First of install NumPy :-

```
pip install numpy
```

Import NumPy

Once NumPy is installed, import it in your applications by adding the `import` keyword:

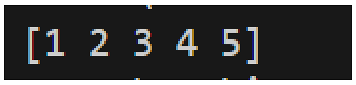
Example 1:-

```python
import numpy as np

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

print(arr)
```

output:-



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

Example

Create a 2-D array containing two arrays with the values 1,2,3 and 4,5,6:

```
import numpy as np

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

print(arr)
```

output:-

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

3-D arrays

An array that has 2-D arrays (matrices) as its elements is called 3-D array.

These are often used to represent a 3rd order tensor.

Example

Create a 3-D array with two 2-D arrays, both containing two arrays with the values 1,2,3 and 4,5,6:

```
import numpy as np

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

print(arr)
```

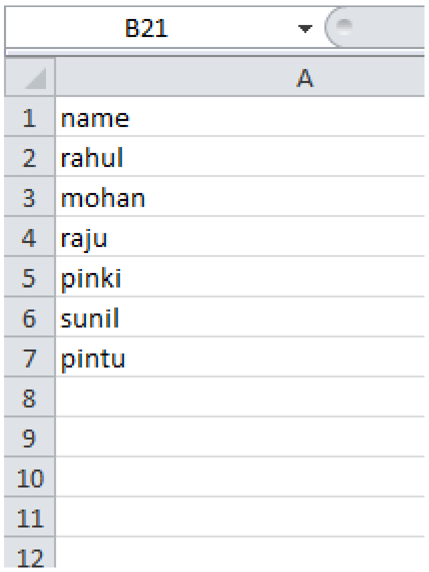
Output:-

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

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

Now reading csv file :-

Cerate csv file book.csv as shown below



B21

	A
1	name
2	rahul
3	mohan
4	raju
5	pinki
6	sunil
7	pintu
8	
9	
10	
11	
12	

Now to read this csv file using NumPy :-

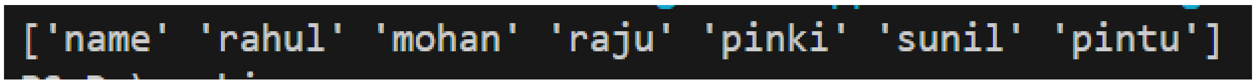
Read CSV files Using NumPy genfromtxt() method

The genfromtxt() method is used to import the data from a text document

```
import numpy as np

# using genfromtxt()
arr = np.genfromtxt("book.csv",
                    delimiter=",", dtype=str)
print(arr)
```

output:-



```
['name' 'rahul' 'mohan' 'raju' 'pinki' 'sunil' 'pintu']
```

Read CSV files Using built-in Python csv module

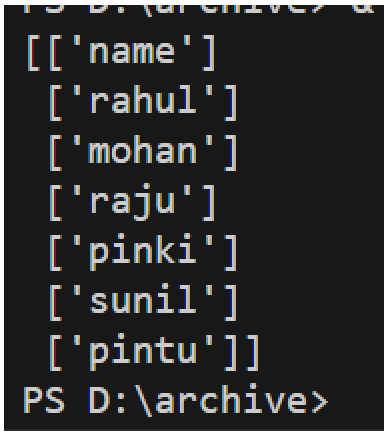
```
import numpy as np

# Importing csv module
import csv


with open("book.csv", 'r') as x:
    sample_data = list(csv.reader(x, delimiter=","))

sample_data = np.array(sample_data)
print(sample_data)
```

output:-



```
[['name']
 ['rahul']
 ['mohan']
 ['raju']
 ['pinki']
 ['sunil']
 ['pintu']]
PS D:\archive>
```

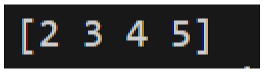
Slice elements from index 1 to index 5 from the following array:

```
import numpy as np

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

print(arr[1:5])
```

output:-



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

Searching Arrays

You can search an array for a certain value, and return the indexes that get a match.

To search an array, use the `where()` method.

Example

Find the indexes where the value is 4:

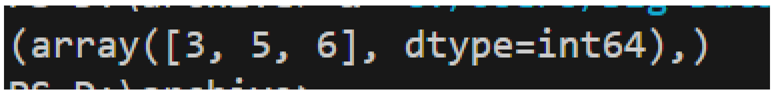
```
import numpy as np

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

x = np.where(arr == 4)

print(x)
```

Output:-



```
(array([3, 5, 6], dtype=int64),)
```

The NumPy ndarray object has a function called `sort()`, that will sort a specified array.

Example

Sort the array:

```
import numpy as np

arr = np.array([3, 2, 0, 1])
```

```
print(np.sort(arr))
```

output:-

```
[0 1 2 3]
```

Filtering Arrays

Getting some elements out of an existing array and creating a new array out of them is called *filtering*.

In NumPy, you filter an array using a *boolean index list*.

Note: If the value at an index is `True` that element is contained in the filtered array, if the value at that index is `False` that element is excluded from the filtered array.

Example 1

Create an array from the elements on index 0 and 2:

```
import numpy as np

arr = np.array([41, 42, 43, 44])

x = arr[[True, False, True, False]]

print(x)
```

Output:-

```
[41 43]
```

Example 2

Create a filter array that will return only values higher than 42:

```
import numpy as np

arr = np.array([41, 42, 43, 44])
```

```
# Create an empty list
filter_arr = []

# go through each element in arr
for element in arr:
  # if the element is higher than 42, set the value to True, otherwise False:
  if element > 42:
    filter_arr.append(True)
  else:
    filter_arr.append(False)

newarr = arr[filter_arr]

print(filter_arr)
print(newarr)
```

output:-

```
[False, False, True, True]
[43 44]
```

Creating Filter Directly From Array

The above example is quite a common task in NumPy and NumPy provides a nice way to tackle it.

We can directly substitute the array instead of the iterable variable in our condition and it will work just as we expect it to.

Example

Create a filter array that will return only values higher than 42:

```
import numpy as np

arr = np.array([41, 42, 43, 44])

filter_arr = arr > 42
```

```
newarr = arr[filter_arr]

print(filter_arr)
print(newarr)
```

output:-

```
[False False  True  True]
[43 44]
```