* Inheritance and Overriding:

31/05/25

Numpy:

→ These are used to store Data in different forms: 1D, 2D, 3D... ND
→ Time efficiency
→ used for working with arrays
→ The array object in numpy is called ndarray.

```
import numpy as np
a = [1,2,3]
a = np.array(a)
a.ndim → (1) → gives dimensions of the array
```

```
a = [[1,2,3],[4,5,6]]
np.array(a) → [[1 2 3]
               [4 5 6]]
a.ndim → (2)
```

* Changing data type of a array:

```
a = np.array(a, dtype = float)
a = np.array(a, dtype = 'str')
```

* To know how many records and columns in a list then

```
a.shape → (3,)
b.shape → (2,2)
c.shape → (3,3)
```

Accessing / Changing specific elements, rows and columns

```
a = np.array([[1,2,3,4,5,6,7],[8,9,10,11,12,13,14]])
print(a)
```

```
O/P: [[1, 2, 3, 4, 5, 6, 7]
      [8, 9, 10, 11, 12 13 14]]
I/P: a[0,3]
O/P: 4
I/P: a[1,4]
O/P: 12
```

```
#Get a specific row from a numpy arr:
I/P: a[0,:]
O/P: arr([1,2,3,4,5,6,7])
#Get a specific column
I/P: a[:,1]
O/P: arr([2,9])
```

→ changing value / Element: a[0,3] = 20 → replaces first row and 4th column with 20.

i/p: np.zeros((3,3))

o/p: array([[0,0,0],
[0,0,0],
[0,0,0]])

i/p: np.ones((3,3))

o/p: array([[1.,1.,1.],
[1.,1.,1.],
[1.,1.,1.]])

i/p: np.ones((3,3), dtype='int')

* using full function: create a full array of the number that we declare

i/p: np.full((3,3), 50, dtype='int')

* if you want to pass a random number then it is:

→ np.random.rand(4,4)

* if you want random integers in a certain/specified range:

i/p: np.random.randint(1,5, size=(3,3))

o/p: array([[3,4,2],
[1,1,3],
[1,2,4]])

* identity function: $\begin{bmatrix}1&0\\0&1\end{bmatrix}$ → np.identity(2)

* Arithematic Operations with Numpy

a = np.ones((3,3))

$a = \begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$

a = a+2 ⇒ $\begin{bmatrix}3&3&3\\3&3&3\\3&3&3\end{bmatrix}$

Similarly, a = a*2 ⇒ $\begin{bmatrix}6&6&6\\6&6&6\\6&6&6\end{bmatrix}$

→ similarly, you can perform All Arithematic Operations.

→ a.min()
→ np.min(a)
→ a.max()
→ a.mean()

Pandas

~~dfate~~ → import pandas as PD

→ pd.read_csv(' ') → reads the data and creates a dataframe.

→ Pd.head() → retrieves the first 5 records

→ Pd.tail() → retrieves the last 5 records

→ Pd.info() → retrieves the information of the dataset/dataframe.

→ Pd.describe() → Describes the dataframe without including categorical data

→ Pd.columns → Retrieves all the column names

→ Pd.T → Returns transpose of the dataframe

→ pd.sort_values('Age') → Returns all the values in Ascending orde for column Age.

→ pd['balance'] = Pd['balance'] +1000 → sums all records of the column balance with 1000.