NumPy and Pandas - Python Libraries Summary

NUMPY (Numerical Python)

NumPy provides support for large multidimensional arrays and mathematical functions.

Import:

    import numpy as np

Array Creation:

    np.array([1, 2, 3])

    np.zeros((2,3))

    np.ones((3,2))

    np.arange(0,10,2)

    np.linspace(0,1,5)

Attributes:

    arr.ndim -> Dimensions

    arr.shape -> Shape (rows, cols)

    arr.size -> Total elements

    arr.dtype -> Data type

Array Operations:

    a + b, a - b, a * b, a / b

Statistical Functions:

```
np.mean(arr), np.median(arr), np.std(arr), np.sum(arr)
```

Indexing and Slicing:

```
arr[0], arr[1:4], arr[:,1]
```

Example:

```
import numpy as np

a = np.array([10,20,30])

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

print(a + b)
```

PANDAS (Python Data Analysis Library)

Pandas provides powerful data structures for data analysis:

- Series (1D labeled array)

- DataFrame (2D table like Excel)

Import:

```
import pandas as pd
```

Series Example:

```
s = pd.Series([10, 20, 30], index=['A','B','C'])

print(s['A'])
```

DataFrame Example:

```python
data = {'Name':['John','Alice'],'Age':[25,30]}

df = pd.DataFrame(data)
```

Accessing Data:

```python
df['Name'], df[['Name','Age']]

df.iloc[0], df.loc[1, 'Age']
```

Adding/Removing Columns:

```python
df['Salary'] = [40000,50000]

df.drop('Age', axis=1, inplace=True)
```

Filtering:

```python
df[df['Salary'] > 40000]
```

Common Functions:

```python
df.head(), df.tail(), df.info(), df.describe(), df.shape

df.isnull(), df.fillna(), df.dropna()

df.sort_values('Salary')

df.groupby('City').mean()
```

File Operations:

```python
df = pd.read_csv('data.csv')

df.to_csv('output.csv', index=False)
```

NUMPY vs PANDAS

Feature	NumPy	Pandas
Structure	ndarray	Series, DataFrame
Data Type	Homogeneous	Heterogeneous
Use Case	Numerical computation	Data analysis/manipulation
Indexing	Position	Label + Position
Speed	Faster for numeric	Slower but flexible

Example Using Both:

```python
import numpy as np
import pandas as pd


arr = np.random.randint(1,100,size=(3,3))
df = pd.DataFrame(arr, columns=['A','B','C'])


print(arr)
print(df)
```