

What is NumPy?

NumPy (Numerical Python) is a Python library used for:

- Fast numerical computations
- Working with **arrays (multidimensional data)**
- Performing **mathematical operations** efficiently

Install it first (if not already installed):

```
pip install numpy
```

Import it:

```
import numpy as np
```

1. Creating NumPy Arrays

From Python lists:

```
import numpy as np
```

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

```
print(a)
```

```
print(type(a))
```

Output:

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

```
<class 'numpy.ndarray'>
```

2D Array:

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

```
print(b)
```

2. Array Attributes

You can get important info about arrays:

```
print(b.ndim) # Number of dimensions
```

```
print(b.shape) # Rows, Columns
```

```
print(b.size) # Total number of elements
```

```
print(b.dtype) # Data type of elements
```

3. Creating Arrays Quickly

Function	Description	Example
<code>np.zeros()</code>	All zeros	<code>np.zeros((2,3))</code>
<code>np.ones()</code>	All ones	<code>np.ones((3,3))</code>
<code>np.full()</code>	All same value	<code>np.full((2,2), 9)</code>
<code>np.eye()</code>	Identity matrix	<code>np.eye(3)</code>
<code>np.arange()</code>	Range of numbers	<code>np.arange(0,10,2)</code>
<code>np.linspace()</code>	Evenly spaced values	<code>np.linspace(0,1,5)</code>
<code>np.random.rand()</code>	Random numbers (0–1)	<code>np.random.rand(2,3)</code>
<code>np.random.randint()</code>	Random integers	<code>np.random.randint(10, 50, (2,3))</code>

Example:

```
arr = np.arange(0, 10, 2)
print(arr)
```

4. Array Operations

NumPy allows **element-wise operations**:

```
x = np.array([1, 2, 3])
y = np.array([4, 5, 6])
```

```
print(x + y) # [5 7 9]
print(x * y) # [ 4 10 18]
print(x ** 2) # [1 4 9]
```

5. Array Functions

Basic Math

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

```
print(np.sum(a))    # 15
print(np.mean(a))   # 3.0
print(np.max(a))    # 5
print(np.min(a))    # 1
print(np.std(a))    # 1.414...
```

Matrix Operations

```
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
```

```
print(np.dot(A, B)) # Matrix multiplication
```

6. Indexing and Slicing

```
arr = np.array([10, 20, 30, 40, 50])
```

```
print(arr[0])    # First element
print(arr[1:4])  # Slice (20,30,40)
print(arr[-1])   # Last element
```

2D slicing:

```
b = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(b[0, 1])   # element at row 0, column 1 (2)
print(b[1:, 1:]) # subarray from 2nd row onward, 2nd column onward
```

7. Reshaping and Flattening

```
a = np.arange(1, 10)
print(a.reshape(3,3)) # Change shape
print(a.flatten())    # Flatten to 1D
```

8. Stacking and Splitting

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

print(np.hstack((a,b))) # Horizontal stack
print(np.vstack((a,b))) # Vertical stack
```

9. Boolean Indexing

```
arr = np.array([10, 20, 30, 40, 50])
print(arr[arr > 25]) # [30 40 50]
```

10. Useful Functions

```
np.sort(arr) # Sort array
np.unique(arr) # Remove duplicates
np.concatenate([a,b])
```

Example: Combine Everything

```
import numpy as np

data = np.random.randint(10, 100, (4, 3))
print("Data:\n", data)

print("Mean per column:", np.mean(data, axis=0))
print("Max per row:", np.max(data, axis=1))
print("Sorted data:\n", np.sort(data))
```