EX.NO:	**BASIC ARRAY OPERATIONS USING NUMPY PACKAGES**
DATE: / /2025	

AIM

To Implement the Basic Array Operations using numpy packages for arrays in python.

SOURCE CODE:

```
import numpy as np
print("\n=== 1. Creating Basic Arrays ===")
arr1 = np.array([1, 2, 3, 4, 5])
arr2 = np.array([[10, 20, 30], [40, 50, 60]])
print("Array 1:", arr1)
print("Array 2:\n", arr2)
print("\n=== 2. Array Properties ===")
print("Shape of arr2:", arr2.shape)
print("Dimensions:", arr2.ndim)
print("Data type:", arr1.dtype)
print("Size of arr2:", arr2.size)
print("\n=== 3. Special Arrays ===")
print("Zeros:\n", np.zeros((2, 3)))
print("Ones:\n", np.ones((3, 2)))
print("Identity:\n", np.eye(4))
print("Range:", np.arange(0, 20, 3))
print("Linspace:", np.linspace(1, 10, 5))
print("\n=== 4. Reshaping ===")
reshaped = np.arange(1, 13).reshape(3, 4)
print("Original 1–12 reshaped to 3x4:\n", reshaped)
print("\n=== 5. Indexing & Slicing ===")
print("First element:", arr1[0])
print("Last two elements:", arr1[-2:])
print("Row 1 of reshaped:", reshaped[1])
print("Element (2,3):", reshaped[2, 3])
print("\n=== 6. Arithmetic Operations ===")
print("arr1 + 10:", arr1 + 10)
print("arr1 * 2:", arr1 * 2)
print("arr2 + arr2:\n", arr2 + arr2)
print("\n=== 7. Vectorized Math Functions ===")
print("Square:", np.square(arr1))
print("Square root:", np.sqrt(arr1))
```

```
print("Exponential:", np.exp(arr1))
print("Sine:", np.sin(arr1))
print("\n=== 8. Aggregate Functions ===")
print("Sum:", np.sum(arr1))
print("Min:", np.min(arr1))
print("Std deviation:", np.std(arr1))
print("\n=== 9. Matrix Operations ===")
matrixA = np.array([[1, 2], [3, 4]])
matrixB = np.array([[5, 6], [7, 8]])
print("Matrix A:\n", matrixA)
print("Matrix B:\n", matrixB)
print("Matrix Addition:\n", matrixA + matrixB)
print("Matrix Multiplication:\n", np.dot(matrixA, matrixB))
print("Transpose of A:\n", matrixA.T)
print("\n=== 10. Random Arrays ===")
print("Random integers:\n", np.random.randint(1, 50, (3, 3)))
print("Random floats:\n", np.random.rand(3, 3))
print("Normal distribution:\n", np.random.randn(3, 3))
print("\n=== 11. Boolean Masking ===")
nums = np.array([5, 10, 15, 20, 25, 30])
print("Numbers:", nums)
print("Greater than 15:", nums[nums > 15])
print("Even numbers:", nums[nums % 2 == 0])
print("\n=== 12. Concatenate Arrays ===")
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
print("Concat 1D:", np.concatenate((a, b)))
c1 = np.array([[1, 2], [3, 4]])
c2 = np.array([[5, 6], [7, 8]])
print("Concat 2D vertically:\n", np.vstack((c1, c2)))
print("Concat 2D horizontally:\n", np.hstack((c1, c2)))
print("\n=== 13. Unique & Sorting ===")
arr = np.array([5, 3, 8, 3, 9, 5, 1])
print("Original:", arr)
print("Sorted:", np.sort(arr))
print("Unique:", np.unique(arr))
print("\n=== 14. Save & Load Arrays ===")
np.save("sample_array.npy", arr1)
loaded = np.load("sample_array.npy")
print("Loaded array:", loaded)

print("\n=== PROGRAM COMPLETED SUCCESSFULLY ===")
```

OUTPUT:

```
=== 1. Creating Basic Arrays ===
Array 1: [1 2 3 4 5] Array 2:
 [[10 20 30]
 [40 50 60]]
=== 2. Array Properties ===
Shape of arr2: (2, 3)
Dimensions: 2
Data type: int64
Size of arr2: 6
=== 3. Special Arrays === Zeros:
 [[0. 0. 0.]
[0. 0. 0.]] Ones:
 [[1. 1.]
[1. 1.]
 [1. 1.]] Identity:
 [[1. 0. 0. 0.]
[0. 0. 0. 1.]]
Range: [ 0  3  6  9 12 15 18]
Linspace: [ 1.    3.25  5.5   7.75 10.  ]
=== 4. Reshaping === Original
1–12 reshaped to 3x4:
 [[ 1  2  3  4]
 [ 9 10 11 12]]
=== 5. Indexing & Slicing ===
First element: 1
Last two elements: [4 5]
Row 1 of reshaped: [5 6 7 8]
Element (2,3): 12
=== 6. Arithmetic Operations ===
arr1 + 10: [11 12 13 14 15] arr1 *
2: [ 2  4  6  8 10] arr2 + arr2:  [[ 20
40  60]
 [ 80 100 120]]
=== 7. Vectorized Math Functions ===
Square: [ 1  4  9 16 25]
Square root: [1.         1.41421356 1.73205081 2.         2.23606798]
Exponential: [  2.71828183   7.3890561   20.08553692  54.59815003 148.4131591 ]
Sine: [ 0.84147098  0.90929743  0.14112001 -0.7568025  -0.95892427]
=== 8. Aggregate Functions ===
Sum: 15
Mean: 3.0
Std deviation: 1.4142135623730951
=== 9. Matrix Operations === Matrix
A:
 [[1 2]
```

```
 [3 4]]
Matrix B:
 [[5 6]
 [7 8]]
Matrix Addition:
 [[ 6  8]
 [10 12]]
Matrix Multiplication:
 [[19 22]
 [43 50]]
Transposeof A:
 [[1 3]
 [2 4]]
=== 10. Random Arrays === Random
integers:
 [[26  6  9]
 [45 29 43]
 [32 19 34]] Random
Normal distribution:
 [[ 1.0872158   0.15802423 -0.372782  ]
=== 11. Boolean Masking ===
Numbers: [ 5 10 15 20 25 30]
Greater than 15: [20 25 30]
Even numbers: [10 20 30]
=== 12. Concatenate Arrays ===
Concat 1D: [1 2 3 4 5 6] Concat
2D vertically:
 [[1 2]
 [3 4]
 [5 6]
 [7 8]]
Concat 2D horizontally:
 [[1 2 5 6]
 [3 4 7 8]]
=== 13. Unique & Sorting ===
Original: [5 3 8 3 9 5 1]
Sorted: [1 3 3 5 5 8 9]
Unique: [1 3 5 8 9]
=== 14. Save & Load Arrays ===
Loaded array: [1 2 3 4 5]
=== PROGRAM COMPLETED SUCCESSFULLY ===
```

RESULT:

The program has been successfully executed.