NUMPY ASSIGNMENT

1. What are Broadcasting Rules?

Broadcasting in NumPy allows operations between arrays of different shapes. The rules are:

1. NumPy compares array shapes from right to left.
2. Two dimensions are compatible if:
    - They are equal, OR
    - One of them is 1.
3. If dimensions are not compatible, an error occurs.

Example:

```
import numpy as np

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

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

print(a + b)
```

2. Why Broadcasting is Important in Machine Learning?

Broadcasting is important because:

- Machine learning relies heavily on matrix and vector operations.
- It removes the need for explicit loops.
- It increases computational efficiency.
- It is used in adding bias terms, feature scaling, and gradient calculations.

3. What is a Random Number?

A random number is a number generated without a fixed pattern. In NumPy, random numbers are generated using the `numpy.random` module.

Example:

```
np.random.rand()
```

4. Difference Between rand() and randint()

rand()	randint()
Generates floating-point numbers	Generates integer numbers
Values between 0 and 1	Values within user-defined range
Continuous values	Discrete values

Example:

```
np.random.rand(2,2)
np.random.randint(1,10,(2,2))
```

5. Explain Sorting in NumPy

Sorting arranges array elements in ascending order by default.

Example:

```
a = np.array([5,2,8,1])
print(np.sort(a))
```

For 2D arrays:

```
np.sort(a, axis=0)
np.sort(a, axis=1)
```

6. How Can We Search Data in NumPy?

Searching can be done using:

- `where()`
- `searchsorted()`
- `nonzero()`

Example:

```
a = np.array([10,20,30,40])
print(np.where(a == 20))
```

7. What is Joining? Explain with Examples

Joining means combining two or more arrays.

Using concatenate()

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

np.concatenate((a,b))
```

Using stack()

```
np.vstack((a,b))
np.hstack((a,b))
```

8. Explain Shape and Reshape

Shape

Returns the dimensions of an array.

```
a.shape
```

Reshape

Changes the structure of an array without changing the data.

```
a.reshape(2,3)
```

9. Difference Between Copy and View

View	Copy
Shares original data	Creates independent data
Changes affect original	Changes do not affect original

Example:

```
a = np.array([1,2,3])
b = a.view()
c = a.copy()
```

10. Difference Between flatten() and ravel()

flatten()	ravel()
Returns a copy	Returns a view (if possible)
Slower	Faster
Changes do not affect original	Changes may affect original

Example:

```
a.flatten()
a.ravel()
```

11. Difference Between Copy and Shallow Copy

Deep Copy	Shallow Copy
Independent memory allocation	Shares memory reference
No effect on original array	Changes may reflect in original

In NumPy: - `copy()` creates a deep copy. - `view()` creates a shallow copy.

.