2) The Shape and Reshaping of NumPy Array

1. Dimensions of a NumPy Array

The number of dimensions (axes) in an array is given by the ndim attribute:

```
import numpy as np

# Creating a 2-D array

array_2d = np.array([[1, 2, 3], [4, 5, 6]])

print(array_2d.ndim)


 # Output: 2
```

2. Shape of a NumPy Array

The shape of an array, representing the size of each dimension, is accessible via the shape attribute:

```
print(array_2d.shape)

 # Output: (2, 3)
```

3. Size of a NumPy Array

The total number of elements in the array is obtained using the size attribute:

```
print(array_2d.size)

# Output: 6
```

4. Reshaping a NumPy Array

To change the shape of an array without altering its data, use the reshape() method. The new shape must be compatible with the original size:

```
array_1d = np.array([1, 2, 3, 4, 5, 6])

array_2d = array_1d.reshape((2, 3))

print(array_2d)

# Output:

# [[1 2 3]

#  [4 5 6]]
```

Alternatively, you can use the np.reshape() function:

```
array_2d = np.reshape(array_1d, (2, 3))

print(array_2d)

# Output:
```

```
# [[1 2 3]
#  [4 5 6]]
```

5. Flattening a NumPy Array

To convert a multi-dimensional array into a 1-D array, use the flatten() method or ravel() function:

```
array_2d = np.array([[1, 2, 3], [4, 5, 6]])
array_1d = array_2d.flatten()
print(array_1d)
 # Output: [1 2 3 4 5 6]
```

6. Transpose of a NumPy Array

To transpose an array (swap rows and columns), use the T attribute:

```
array_2d = np.array([[1, 2, 3], [4, 5, 6]])
array_transposed = array_2d.T
print(array_transposed)
# Output:
# [[1 4]
#  [2 5]
#  [3 6]]
```