1d, 2d, 3d, 4d arrays indexing:

1d array:

A 1D array is the simplest type of array, containing elements arranged in a single dimension.
It behaves like a list of items in Python but is more efficient for numerical computations when using libraries like NumPy.
Characteristics of a 1D Array:
Linear Structure: A single row or a single column of elements. No additional dimensions for grouping.
Shape: Represented as (n,), where n is the number of elements.For example, an array with 5 elements has the shape (5,).
Indexing: Access elements using a single index, like array[index].

2d array:

A 2D array is a data structure that organizes data in two dimensions: rows and columns.
It can be visualized as a table or grid, where each cell contains an element, and you can access each element using two indices:
one for the row and one for the column.
Characteristics of a 2D Array:
Rows: Horizontal collections of elements.
Columns: Vertical collections of elements.
Shape: Represented as (m, n), where: m = Number of rows, n = Number of columns.
Indexing: Use two indices to access elements: array[row, column].

3d array:

A 3D array is a data structure that organizes data in three dimensions. It is like a stack of 2D arrays (matrices),
where each 2D array represents a "slice" or a "layer." You can access elements in a 3D array using three indices:
one for the layer, one for the row, and one for the column.
Characteristics of a 3D Array:

Layers, Rows, and Columns:
Layers: The outermost dimension, representing a collection of 2D arrays.
Rows: The horizontal elements in each 2D slice.
Columns: The vertical elements in each 2D slice.
Shape: Represented as (x, y, z), where:
x = Number of layers (depth).
y = Number of rows in each layer.
z = Number of columns in each layer.
Indexing: Use three indices to access elements: array[layer, row, column].

4d array:
A 4D array is a data structure that organizes data in four dimensions.
It can be visualized as a collection of 3D arrays stacked together, where each "3D block" contains multiple layers, rows, and columns.
You can think of it as a "stack of cubes", and you use four indices to access an element: one for the outermost stack (4th dimension),
one for the layer (3rd dimension), one for the row (2nd dimension), and one for the column (1st dimension).
Visual Representation of a 4D Array:
For the array with shape (2, 3, 2, 4):
2 blocks (4th dimension). Each block contains 3 layers (3rd dimension).
Each layer has 2 rows (2nd dimension).Each row contains 4 elements (1st dimension).

1d array indexing:
```
import numpy as np
# indexing:
array_1d = np.array([10, 20, 30, 40, 50])
element_1 = array_1d[0]
element_2 = array_1d[1]
element_3 = array_1d[2]
element_4 = array_1d[-1]
element_5 = array_1d[-2]
```

```
print("element at 0 index is:", element_1)
print("element at 1 index is:", element_2)
print("element at 2 index is:", element_3)
print("element at -1 index is:", element_4)
print("element at -2 index is:", element_5)
```

Output:

```
element at 0 index is: 10
element at 1 index is: 20
element at 2 index is: 30
element at -1 index is: 50
element at -2 index is: 40
```

 # 2d array indexing:

```
import numpy as np
array_2d = np.array([
    [10, 20, 30, 40],
    [50, 60, 70, 80],
    [90, 100, 110, 120]
])
# Indexing:
element_1 = array_2d[0,1]
element_2 = array_2d[1,2]
element_3 = array_2d[2,3]
element_4 = array_2d[0,2]
element_5 = array_2d[1,0]
element_6 = array_2d[2,1]

print("element 1 is:", element_1)
print("element 2 is:", element_2)
print("element 3 is:", element_3)
print("element 4 is:", element_4)
print("element 5 is:", element_5)
print("element 6 is:", element_6)
```

Output:

```
element 1 is: 20
element 2 is: 70
element 3 is: 120
element 4 is: 30
element 5 is: 50
element 6 is: 100
```

 # 3d array indexing:

```
import numpy as np
array_3d = np.array([
    [[10,20,30],[40,50,60],[70,80,90]],
    [[100,110,120],[130,140,150],[160,170,180]],
    [[190,200,210],[220,230,240],[250,260,270]]
])
# indexing:
element_1 = array_3d[0,1,2]
element_2 = array_3d[1,1,0]
element_3 = array_3d[2,2,1]
element_4 = array_3d[0,0,2]
element_5 = array_3d[1,2,1]
element_6 = array_3d[2,1,0]

print("element 1 is:", element_1)
print("element 2 is:", element_2)
print("element 3 is:", element_3)
print("element 4 is:", element_4)
print("element 5 is:", element_5)
print("element 6 is:", element_6)
```

Output:

```
element 1 is: 60
element 2 is: 130
element 3 is: 260
element 4 is: 30
element 5 is: 170
element 6 is: 220
```

 # 4d array indexing:

```
import numpy as np
array_4d = np.array([
    [
    [[10,20],[30,40]],
    [[50,60],[70,80]],
    [[90,100],[110,120]]
    ],
    [
        [[130,140],[150,160]],
        [[170,180],[190,200]],
        [[210,220],[230,240]]
    ]
])
# indexing:
element_1 = array_4d[0,0,0,0]
element_2 = array_4d[0,1,1,1]
element_3 = array_4d[1,1,1,1]
element_4 = array_4d[1,0,0,0]
element_5 = array_4d[0,0,0,1]
element_6 = array_4d[1,1,0,0]

print("element 1 is:", element_1)
print("element 2 is:", element_2)
print("element 3 is:", element_3)
print("element 4 is:", element_4)
```

```
print("element 5 is:", element_5)
print("element 6 is:", element_6)
```

Output:

```
element 1 is: 10
element 2 is: 80
element 3 is: 200
element 4 is: 130
element 5 is: 20
element 6 is: 170
```