NumPy Array Slicing

Array Slicing is the process of extracting a portion of an array.

With slicing, we can easily access elements in the array. It can be done on one or more dimensions of a NumPy array.

Syntax of NumPy Array Slicing

Here's the syntax of array slicing in NumPy:

array[start:stop:step]

Here,

- start - index of the first element to be included in the slice
- stop - index of the last element (exclusive)
- step - step size between each element in the slice

Note: When we slice arrays, the start index is inclusive but the stop index is exclusive.

- If we omit start, slicing starts from the first element
- If we omit stop, slicing continues up to the last element
- If we omit step, default step size is 1

1D NumPy Array Slicing

In NumPy, it's possible to access the portion of an array using the slicing operator :. For example,

```
import numpy as np
# create a 1D array
array1 = np.array([1, 3, 5, 7, 8, 9, 2, 4, 6])
# slice array1 from index 2 to index 6 (exclusive)
print(array1[2:6])  # [5 7 8 9]
# slice array1 from index 0 to index 8 (exclusive) with a step size of 2
print(array1[0:8:2])  # [1 5 8 2]
# slice array1 from index 3 up to the last element
print(array1[3:])  # [7 8 9 2 4 6]
# items from start to end
print(array1[:])   # [1 3 5 7 8 9 2 4 6]
```

In the above example, we have created the array named array1 with **9** elements.

Then, we used the slicing operator : to slice array elements.

- array1[2:6] - slices array1 from index **2** to index **6**, not including index **6**
- array1[0:8:2] - slices array1 from index **0** to index **8**, not including index **8**
- array1[3:] - slices array1 from index **3** up to the last element
- array1[:] - returns all items from beginning to end

Modify Array Elements Using Slicing

With slicing, we can also modify array elements using:

- start parameter
- stop parameter
- start and stop parameter
- start, stop, and step parameter

1. Using start Parameter

```
import numpy as np
# create a numpy array
numbers = np.array([2, 4, 6, 8, 10, 12])
# modify elements from index 3 onwards
numbers[3:] = 20
print(numbers)

# Output: [ 2  4  6 20 20 20]
```

Here, numbers[3:] = 20 replaces all the elements from index **3** onwards with new value **20**.

2. Using stop Parameter

```
import numpy as np

# create a numpy array
numbers = np.array([2, 4, 6, 8, 10, 12])

# modify the first 3 elements
numbers[:3] = 40
print(numbers)

# Output: [40 40 40  8 10 12]
```

Here, numbers[:3] = 20 replaces the first 3 elements with the new value **40**.

3. Using start and stop parameter

```
import numpy as np
# create a numpy array
numbers = np.array([2, 4, 6, 8, 10, 12])
# modify elements from indices 2 to 5
numbers[2:5] = 22
print(numbers)

# Output: [2 4 22 22 22 12]
```

Here, numbers[2:5] = 22 selects elements from index **2** to index **4** and replaces them with new value **22**.

4. Using start, stop, and step parameter

```
import numpy as np
# create a numpy array
numbers = np.array([2, 4, 6, 8, 10, 12])
# modify every second element from indices 1 to 5
numbers[1:5:2] = 16
print(numbers)

# Output: [ 2 16  6 16 10 12]
```

In the above example,

numbers[1:5:2] = 16

modifies every second element from index **1** to index **5** with a new value **16**.

NumPy Array Negative Slicing

We can also use *negative indices* to perform negative slicing in NumPy arrays. During negative slicing, elements are accessed from the end of the array.

Let's see an example.

```
import numpy as np
# create a numpy array
numbers = np.array([2, 4, 6, 8, 10, 12])
# slice the last 3 elements of the array
```

```
# using the start parameter
print(numbers[-3:])    # [8 10 12]
# slice elements from 2nd-to-last to 4th-to-last
element
# using the start and stop parameters
print(numbers[-5:-2])    # [4 6 8]
# slice every other element of the array from the
end
# using the start, stop, and step parameters
print(numbers[-1::-2])   # [12 8 4]
```

Output

```
Using numbers[-3:]- [ 8 10 12]
Using numbers[-5:-2]- [4 6 8]
Using numbers[-1::-2]- [12  8  4]
```

Here,

- numbers[-3:] - slices last 3 elements of numbers
- numbers[-5:-2] - slices numbers elements from 5th last to 2nd last(excluded)
- numbers[-1::-2] - slices every other numbers elements from the end with step size **2**

Reverse NumPy Array Using Negative Slicing

In NumPy, we can also reverse array elements using the negative slicing. For example,

```
import numpy as np
# create a numpy array
numbers = np.array([2, 4, 6, 8, 10, 12])
# generate reversed array
reversed_numbers = numbers[::-1]
print(reversed_numbers)

# Output: [12 10 8 6 4 2]
```

Here, the slice numbers[::-1] selects all the elements of the array with a step size of **-1**, which reverses the order of the elements.

2D NumPy Array Slicing

A 2D NumPy array can be thought of as a matrix, where each element has two indices, row index and column index.

To slice a 2D NumPy array, we can use the same syntax as for slicing a 1D NumPy array. The only difference is that we need to specify a slice for each dimension of the array.

Syntax of 2D NumPy Array Slicing

```
array[row_start:row_stop:row_step,
col_start:col_stop:col_step]
```

Here,

- row_start,row_stop,row_step - specifies starting index, stopping index, and step size for the rows respectively
- col_start,col_stop,col_step - specifies starting index, stopping index, and step size for the columns respectively

Let's understand this with an example.

```
# create a 2D array
array1 = np.array([[1, 3, 5, 7],
                   [9, 11, 13, 15]])
print(array1[:2, :2])

# Output
[[ 1  3]
 [ 9 11]]
```

Here, the , in [:2, :2] separates the rows of the array.

The first :2 returns first 2 rows i.e., entire array1. This results in

```
[1  3]
```

The second :2 returns first 2 columns from the 2 rows. This results in

```
[9 11]
```

Example: 2D NumPy Array Slicing

```
import numpy as np
# create a 2D array
array1 = np.array([[1, 3, 5, 7],
                   [9, 11, 13, 15],
                   [2, 4, 6, 8]])


# slice the array to get the first two rows and
columns
subarray1 = array1[:2, :2]
# slice the array to get the last two rows and
columns
subarray2 = array1[1:3, 2:4]
# print the subarrays
print("First Two Rows and Columns:
\n",subarray1)
print("Last two Rows and Columns: \n",subarray2)
```

Output

```
First Two Rows and Columns:
[[ 1  3]
 [ 9 11]]
Last two Rows and Columns:
 [[13 15]
 [ 6  8]]
```

Here,

- array1[:2, :2] - slices array1 that starts at the first row and first column (default values), and ends at the second row and second column (exclusive)
- array1[1:3, 2:4] - slices array1 that starts at the second row and third column (index **1** and **2**), and ends at the third row and fourth column (index **2** and **3**)