NumPy - Indexing & Slicing

Contents of ndarray object can be accessed and modified by indexing or slicing, just like Python's in-built container objects.

As mentioned earlier, items in ndarray object follows zero-based index. Three types of indexing methods are available – **field access, basic slicing** and **advanced indexing**.

Basic slicing is an extension of Python's basic concept of slicing to n dimensions. A Python slice object is constructed by giving **start, stop**, and **step** parameters to the built-in **slice** function. This slice object is passed to the array to extract a part of array.

Example 1

Live Demo

```
import numpy as np

a = np.arange(10)

s = slice(2,7,2)

print a[s]
```

Its output is as follows –

```
[2  4  6]
```

In the above example, an **ndarray** object is prepared by **arange()** function. Then a slice object is defined with start, stop, and step values 2, 7, and 2 respectively. When this slice object is passed to the ndarray, a part of it starting with index 2 up to 7 with a step of 2 is sliced.

The same result can also be obtained by giving the slicing parameters separated by a colon : (start:stop:step) directly to the **ndarray** object.

Example 2

Live Demo

```
import numpy as np

a = np.arange(10)

b = a[2:7:2]

print b
```

Here, we will get the same output –

```
[2  4  6]
```

If only one parameter is put, a single item corresponding to the index will be returned. If a : is inserted in front of it, all items from that index onwards will be extracted. If two parameters (with : between them) is used, items between the two indexes (not including the stop index) with default step one are sliced.

Example 3

Live Demo

```
# slice single item

import numpy as np



a = np.arange(10)

b = a[5]

print b
```

Its output is as follows –

```
5
```

Example 4

Live Demo

```
# slice items starting from index

import numpy as np

a = np.arange(10)

print a[2:]
```

Now, the output would be –

```
[2  3  4  5  6  7  8  9]
```

Example 5

Live Demo

```
# slice items between indexes

import numpy as np

a = np.arange(10)

print a[2:5]
```

Here, the output would be –

```
[2  3  4]
```

The above description applies to multi-dimensional **ndarray** too.

Example 6

Live Demo

```
import numpy as np

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

print a



# slice items starting from index

print 'Now we will slice the array from the index a[1:]'

print a[1:]
```

The output is as follows –

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

Now we will slice the array from the index a[1:]
[[3 4 5]
 [4 5 6]]
```

Slicing can also include ellipsis (…) to make a selection tuple of the same length as the dimension of an array. If ellipsis is used at the row position, it will return an ndarray comprising of items in rows.

Example 7

Live Demo

```
# array to begin with

import numpy as np

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


print 'Our array is:'

print a

print '\n'


# this returns array of items in the second column

print 'The items in the second column are:'

print a[...,1]

print '\n'


# Now we will slice all items from the second row

print 'The items in the second row are:'

print a[1,...]

print '\n'


# Now we will slice all items from column 1 onwards

print 'The items column 1 onwards are:'

print a[...,1:]
```

The output of this program is as follows −

```
Our array is:
[[1 2 3]
 [3 4 5]
 [4 5 6]]

The items in the second column are:
[2 4 5]

The items in the second row are:
[3 4 5]

The items column 1 onwards are:
[[2 3]
 [4 5]
 [5 6]]
```