Numpy Array Slicing

Summary: in this tutorial, you'll learn about the numpy array slicing that extracts one or more elements from a numpy array.

Numpy array slicing on on-dimensional arrays

NumPy arrays use brackets `[]` and `:` notations for slicing like lists (https://www.pythontutorial.net/advanced-python/python-slicing/) . By using slices, you can select a range of elements in an array with the following syntax:

```
[m:n]
```

This slice selects elements starting with `m` and ending with `n-1`. Note that the nth element is not included. In fact, the slice `m:n` can be explicitly defined as:

```
[m:n:1]
```

The number 1 specifies that the slice selects every element between `m` and `n`.

To select every two elements, you can use the following slice:

```
[m:n:2]
```

In general, the following expression selects every `k` element between `m` and `n`:

```
[m:n:k]
```

If `k` is negative, the slice returns elements in reversed order starting from `m` to `n+1`. The following table illustrates the slicing expressions:

Slicing Expression	**Meaning**
`a[m:n]`	Select elements with an index starting at m and ending at n-1.
`a[:]` or `a[0:-1]`	Select all elements in a given axis

Slicing Expression	Meaning
`a[:n]`	Select elements starting with index 0 and up to element with index n-1
`a[m:]`	Select elements starting with index m and up to the last element
`a[m:-1]`	Select elements starting with index m and up to the last element
`a[m:n:k]`	Select elements with index m through n (exclusive), with an increment k
`a[::-1]`	Select all elements in reverse order

See the following example:

```
import numpy as np

a = np.arange(0, 10)

print('a=', a)
print('a[2:5]=', a[2:5])
print('a[:]=', a[:])
print('a[0:-1]=', a[0:-1])
print('a[0:6]=', a[0:6])
print('a[7:]=', a[7:])
print('a[5:-1]=', a[5:-1])
```

```
print('a[0:5:2]=', a[0:5:2])
print('a[::-1]=', a[::-1])
```

Output:

```
a= [0 1 2 3 4 5 6 7 8 9]
a[2:5]= [2 3 4]
a[:]= [0 1 2 3 4 5 6 7 8 9]
a[0:-1]= [0 1 2 3 4 5 6 7 8]
a[0:6]= [0 1 2 3 4 5]
a[7:]= [7 8 9]
a[5:-1]= [5 6 7 8]
a[0:5:2]= [0 2 4]
a[::-1]= [9 8 7 6 5 4 3 2 1 0]
```

Numpy array slicing on multidimensional arrays

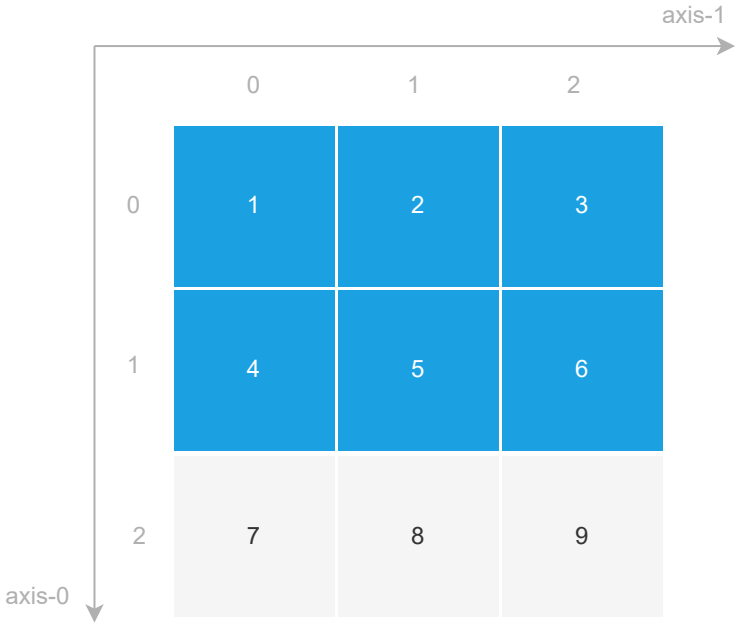
To slice a multidimensional array, you apply the square brackets `[]` and the `:` notation to each dimension (or axis). The slice returns a reduced array where each element matches the selection rules. For example:

```
import numpy as np

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

```
    [4, 5, 6],
    [7, 8, 9]
])


print(a[0:2, :])
```

Output:

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



In this example, array a is a 2-D array. In the expression `a[0:2, :]`:

First, the `0:2` selects the element at index 0 and 1, not 2 that returns:

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

Then, the `:` select all elements. Therefore the whole expression returns:

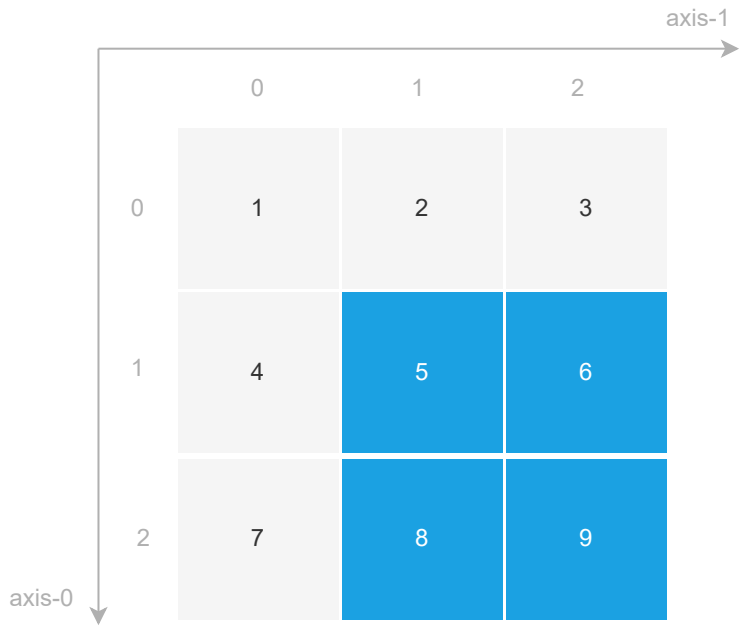
```
[[1 2 3]
 [4 5 6]]
```

Consider another example:

```
import numpy as np

a = np.array([
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
])

print(a[1:, 1:])
```

Output:

```
[[5 6]
 [8 9]]
```



In the expression `a[1:, 1:]`:

First, `1:` selects the elements starting at index 1 to the last element of the first axis (or row), which returns:

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

Second, `1:` selects the elements starting at index 1 to the last elements of the second axis (or column), which returns:

```
[[5 6]
 [8 9]]
```

Summary

- Use slicing to extract elements from a numpy array
- Use `a[m:n:p]` to slice one-dimensional arrays.
- Use `a[m:n:p, i:j:k, ...]` to slice multidimensional arrays