C26 - Numpy

author & date

- author: Akshar Patel
- date: 4/18/2022

Creating an array

```
import numpy as np
```

```
a1 = np.array([1, 2, 3]) #1D array
```

```
a1
```

```
     array([1, 2, 3])
```

```
b2 = np.array([[1, 2, 3], [4, 5, 6]]) #2D array
```

```
b2
```

```
     array([[1, 2, 3],
            [4, 5, 6]])
```

Dimension and size check

```
a1.ndim #number of dimension
```

```
     1
```

```
b2.ndim
```

```
     2
```

```
a1.size #number of elements
```

```
3
```

```
b2.size
```

```
6
```

```
a1.shape #1D array with three elements
```

```
(3,)
```

```
b2.shape #2D array with two rows and three columns
```

```
(2, 3)
```

Easy creation

```
np.zeros(5) # five zeros
```

```
array([0., 0., 0., 0., 0.])
```

```
np.zeros((5,3))
```

```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

```
np.ones(5) #five ones
```

```
array([1., 1., 1., 1., 1.])
```

```
np.ones((5,3))
```

```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

```
np.random.random(5) #five random numbers
```

```
array([0.30536065, 0.22973528, 0.66966663, 0.89357192, 0.63929042])
```

```
np.random.random((5, 3))
```

```
array([[0.97896538, 0.57553229, 0.2160487 ],
       [0.388458  , 0.02157135, 0.54227384],
       [0.24268211, 0.68596133, 0.46000023],
       [0.09419877, 0.62279457, 0.06638199],
       [0.41548602, 0.12113434, 0.75573921]])
```

```
np.arange(5) #from zero to four
```

```
array([0, 1, 2, 3, 4])
```

```
np.arange(2, 9, 2) #(first number, last number, step size)
```

```
array([2, 4, 6, 8])
```

```
np.linspace(0, 10, num =5) #values spaced linearly in a specified interval
```

```
array([ 0. ,  2.5,  5. ,  7.5, 10. ])
```

Sorting and adding

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

```
np.sort(arr) #sort the numbers in ascending order
```

```
array([1, 2, 3, 4, 5, 6, 7, 8])
```

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

```
b = np.array([100, 200, 300, 400])
```

```
np.concatenate((a, b)) #concatenate
```

```
array([  1,   2,   3,   4, 100, 200, 300, 400])
```

indexing and slicing

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

```
data[0]
```

```
1
```

```
data[0:4]
```

```
array([1, 2, 3, 4])
```

```
data2 = np.array([[1, 2, 3], [4, 5, 6]])
```

```
data2[0] #the first item of data2
```

```
array([1, 2, 3])
```

```
data2[0][0] #the first item of the first item of data2
```

```
1
```

```
data3 = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
```

```
data3[data3<7] #all the values less than seven
```

```
array([1, 2, 3, 4, 5, 6])
```

```
data3[data3 % 2 == 0] #all the values that are divisible by 2
```

```
array([ 2,  4,  6,  8, 10, 12])
```

```
data3[(data3 > 11) | (data3 < 3)] #all the values greater than 11 or less than 3
```

```
array([ 1,  2, 12])
```

```
data3[(data3 > 9) & (data3 < 12)] #all the values greater than 9 and less than 12
```

```
array([10, 11])
```

Transposing and reshaping

```
data2 #2 rows, 3 cols
```

```
array([[1, 2, 3],
       [4, 5, 6]])
```

```
data2.reshape(3, 2) #3 rows, 2 cols
```

```
array([[1, 2],
       [3, 4],
       [5, 6]])
```

```
data2.transpose() #compare with the original data2
```

```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

```
data2.T #another way of transposing
```

```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

```
data3 #2D array with 3 rows & 4 cols
```

```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

```
data3.flatten() #from 2D array to 1D array
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

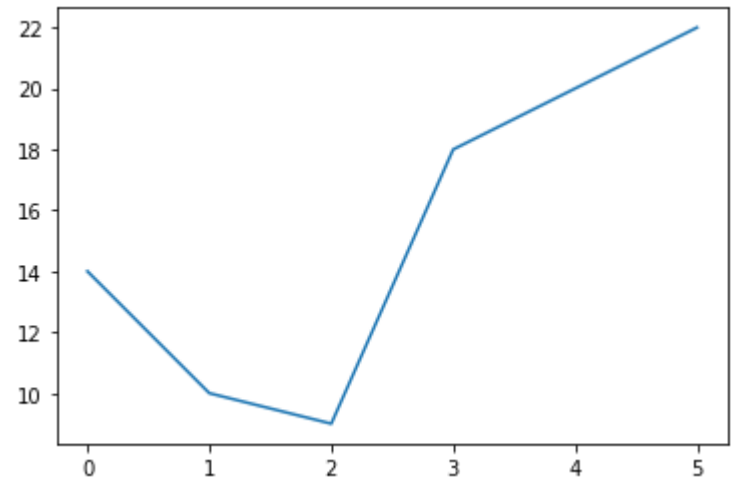
Visualization

```
import matplotlib.pyplot as plt
```

```
score = np.array([14, 10, 9, 18, 20, 22])
```

```
plt.plot(score)
```

```
[<matplotlib.lines.Line2D at 0x7f5f407ccc50>]
```



✓ 0s completed at 6:18 PM