Numpy review homework

1. Make a numpy matrix from a Python list of lists...

In [4]:
```
import numpy as np

list_of_lists = [[1.1, 2.3, 4.6], [10, 15, 3.6], [23, 7.61, 1.01], [9.99, 3, 5.5]]
my_matrix = np.array(list_of_lists)
print(my_matrix)
```

```
[[ 1.1   2.3   4.6 ]
 [10.   15.    3.6 ]
 [23.    7.61  1.01]
 [ 9.99  3.    5.5 ]]
```

2. Make a 3D numpy matrix from a Python list of lists of lists!

In [5]:
```
list_of_lists_of_lists = [[[1,2,3], [4,5,6], [7,8,9]],
                          [[10,11,12], [13,14,15], [16,17,18]],
                          [[19,20,21], [21,22,23], [24,25,26]]]
my_3d_matrix = np.array(list_of_lists_of_lists)
print(my_3d_matrix)

```

```
[[[ 1  2  3]
  [ 4  5  6]
  [ 7  8  9]]

 [[10 11 12]
  [13 14 15]
  [16 17 18]]

 [[19 20 21]
  [21 22 23]
  [24 25 26]]]
```

3. Create a 5x3 array of Gaussian random numbers.

In [6]:
```
random_array = np.random.randn(5, 3)
print(random_array)
```

```
[[ 1.10274804  1.02527696  1.27226392]
 [-1.8119091  -0.52678092  0.56378341]
 [-0.72899902  0.67061466 -0.40828471]
 [-0.84152663 -0.79987753 -0.20720702]
 [ 1.571527   -0.2295595   0.82470127]]
```

4. Write a script to go through the array created in 3. and announce (print) the value and its row and column indexes.

Hint: Use nested `for` loops - one to loop through the rows and one to loop through the columns.

In [11]:
```
rows, columns = random_array.shape

for i in range(rows) :
    for j in range(columns) :
        print(f'This is the value in row {i + 1} and column {j + 1}: {random_array[i, j]}')
```

```
This is the value in row 1 and column 1: 1.1027480393635682
This is the value in row 1 and column 2: 1.0252769563736714
This is the value in row 1 and column 3: 1.2722639197187569
This is the value in row 2 and column 1: -1.8119091000974974
This is the value in row 2 and column 2: -0.5267809197695899
This is the value in row 2 and column 3: 0.5637834105761039
This is the value in row 3 and column 1: -0.7289990201682743
This is the value in row 3 and column 2: 0.670614656921107
This is the value in row 3 and column 3: -0.40828470975708053
This is the value in row 4 and column 1: -0.8415266319864865
This is the value in row 4 and column 2: -0.799877534146249
This is the value in row 4 and column 3: -0.2072070209863994
This is the value in row 5 and column 1: 1.5715270022586088
This is the value in row 5 and column 2: -0.2295594962780532
This is the value in row 5 and column 3: 0.8247012699215408
```

5. Make an new array out of your random numbers such that the mean is 10 and the standard deviation is 3.

In [12]:
```
new_array = random_array * 3 + 10
print(new_array)
print(f'This is the mean of my new array: {new_array.mean()}')
print(f'This is the standard deviation of my new array: {new_array.std()}')

```

```
[[13.30824412 13.07583087 13.81679176]
 [ 4.5642727   8.41965724 11.69135023]
 [ 7.81300294 12.01184397  8.77514587]
 [ 7.4754201   7.6003674   9.37837894]
 [14.71458101  9.31132151 12.47410381]]
This is the mean of my new array: 10.295354164388744
This is the standard deviation of my new array: 2.8312720069740713
```

6. Count the number of values in your new array that are below 7.

In [18]:
```
below_7 = new_array < 7
print(below_7)
print(f'The is {below_7.sum()} values below 7.')

```

```
[[False False False]
 [ True False False]
 [False False False]
 [False False False]
 [False False False]]
The is 1 values below 7.
```

7. Make a numpy sequence that has the even numbers from 2 up to (and including) 20.

In [24]:
```
np.arange(2, 21, 2)
```

Out[24]:
```
array([ 2,  4,  6,  8, 10, 12, 14, 16, 18, 20])
```

8. Get the second and third rows of your array.

In [25]:
```
new_array[1:3, :]
```

Out[25]:
```
array([[ 4.5642727 ,  8.41965724, 11.69135023],
       [ 7.81300294, 12.01184397,  8.77514587]])
```

9. Compute the mean of the columns of your array.

In [26]:
```
new_array.mean(0)
```

Out[26]:
```
array([ 9.57510417, 10.0838042 , 11.22715412])
```