Numpy review homework

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

In [3]:
```
import numpy as np
```

In [6]:
```
numpslist = [[1.2, 2.3, 3.4, 4.5], [1.3,2.4,3.5,4.6], [1.4,2.5,3.6,4.7], [1.5,2.6,3.7,
anothermatrix = np.array(numpslist)
```

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

In [8]:
```
numpslist
```

Out[8]:
```
[[1.2, 2.3, 3.4, 4.5],
 [1.3, 2.4, 3.5, 4.6],
 [1.4, 2.5, 3.6, 4.7],
 [1.5, 2.6, 3.7, 4.8]]
```

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

In [11]:
```
randomly = np.random.randn(5,3)
print(randomly)
```

```
[[ 1.76175652  0.12367195 -0.19305164]
 [-1.47324704  0.98224129 -0.82755152]
 [ 0.33569089  0.7079573   0.38112761]
 [ 0.40138605 -0.98644505 -0.33143813]
 [-1.78121689  0.0811682  -0.79618247]]
```

1. 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 [14]:
```
randomly[0,2]
```

Out[14]:
```
-0.19305163508648487
```

In [62]:
```
rows, cols = randomly.shape

for a in range(rows):
    for b in range(cols):
        print(f"Value: {randomly[a, b]}, Row: {a}, Column: {b}")
```

```
Value: 1.761756524041944, Row: 0, Column: 0
Value: 0.12367195081193122, Row: 0, Column: 1
Value: -0.19305163508648487, Row: 0, Column: 2
Value: -1.4732470374056654, Row: 1, Column: 0
Value: 0.9822412881731639, Row: 1, Column: 1
Value: -0.8275515190454872, Row: 1, Column: 2
Value: 0.3356908926541821, Row: 2, Column: 0
Value: 0.7079573047092321, Row: 2, Column: 1
Value: 0.3811276051322985, Row: 2, Column: 2
Value: 0.4013860529849249, Row: 3, Column: 0
Value: -0.986445047331857, Row: 3, Column: 1
Value: -0.3314381332405502, Row: 3, Column: 2
Value: -1.7812168918901488, Row: 4, Column: 0
Value: 0.0811681951551125, Row: 4, Column: 1
Value: -0.7961824657668916, Row: 4, Column: 2
```

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

In [55]:
```
mean_org = np.mean(randomly)
std_org = np.std(randomly)

wantedmean = 10
wantedstd = 3

adjustedlist = [(num - mean_org) * (wantedstd / std_org) + wantedmean for num in randc

adjustedarray = np.array(adjustedlist)

print("Adjusted Array with Mean 10 and Standard Deviation 3:")
print(adjustedarray)
```

```
Adjusted Array with Mean 10 and Standard Deviation 3:
[[16.1142862  10.75646906  9.72053533]
 [ 5.53329556 13.56466199  7.64522469]
 [11.44993675 12.66753894 11.59855032]
 [11.66481128  7.12551863  9.2679033 ]
 [ 4.52599338 10.61744855  7.74782601]]
```

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

In [56]:
```
below7 = (adjustedarray <7).sum()
print(below7)
```

```
2
```

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

In [57]:
```
even_numbers = np.arange(2, 21, 2)

print("Even numbers from 2-20 :")
print(even_numbers)
```

```
Even numbers from 2-20 :
[ 2  4  6  8 10 12 14 16 18 20]
```

1. Get the second and third rows of your array created in #5.

In [58]:
```
print(adjustedarray[1:3])
```

```
[[ 5.53329556 13.56466199  7.64522469]
 [11.44993675 12.66753894 11.59855032]]
```

1. Compute the mean of the columns of your array created in #5.

In [59]:
```
column_means = np.mean(adjustedarray, axis=0)

print("Mean of the columns:")
print(column_means)
```

```
Mean of the columns:
[ 9.85766464 10.94632743  9.19600793]
```