Getting Started with NumPy - Lab

Introduction

Now that we have introduced NumPy, let's put it to practice. In this lab, we are going to be creating arrays, performing operations on them, and returning new array all using the NumPy library. Let's get started!

Objectives

You will be able to:

- Understand how to initialize NumPy arrays from nested Python lists, and access elements using square brackets
- Understand the shape attribute on NumPy arrays
- Understand how to create arrays from scratch including np.zeros, np.ones, np.full
- Learn to perform scalar and vector math

Import NumPy under the standard alias

In [1]:
```
#Your code here
import numpy as np
```

Generating Some Mock Data

Create a NumPy Array for each of the following:

```
1. Using a range
2. Using a Python List
```

Below, create a list in Python that has 5 elements (i.e. [0,1,2,3,4]) and assign it to the variable `py_list`.

Next, do the same, but instead of a list, create a range with 5 elements and assign it to the variable, `py_range`.

Finally, use the list and range to create NumPy arrays and assign the array from list to the variable `array_from_list`, and the array from the range to the variable `array_from_range`.

In [2]:
```
py_list = [0,1,2,3,4]
py_range = range(0,5)
array_from_list = np.array(py_list)
array_from_range = np.array(py_range)
```

Next, we have a list of heights and weights and we'd like to use them to create a collection of BMIs. However, they are both in inches and pounds (imperial system), respectively.

Let's use what we know to create NumPy arrays with the metric equivalent values, (height in meters & weight in kg).

> **Remember:** *NumPy can make these calculations a lot easier and with less code than a list!*

> 1.0 inch = 0.0254 meters

> 2.2046 lbs = 1 kilogram

In [3]:
```
# use the conversion rate for turning height in inches to meters
list_height_inches = [65, 68, 73, 75, 78]

#Your code here
array_height_inches = np.array(list_height_inches)
array_height_meters = array_height_inches * 0.0254
```

In [4]:
```
# use the conversion rate for turning weight in pounds to kilograms
list_weight_pounds = [150, 140, 220, 205, 265]

#your code here
array_weight_pounds = np.array(list_weight_pounds)
array_weight_kg = array_weight_pounds / 2.2046
```

The metric formula for calculating BMI is as follows:

> BMI = weight (kg) ÷ height^2 (m^2)

So, to get BMI we divide weight by the squared value of height. For example, if i weighed 130kg and was 1.9 meters tall, the calculation would look like:

> BMI = 130 / (1.9*1.9)

Use the BMI calculation to create a NumPy array of BMIs

In [5]:
```
BMI_array = array_weight_kg / (array_height_meters * array_height_meters)
```

Create an identity vector using np.ones()

```
In [6]: #Your code here
        identity = np.ones(len(BMI_array))
        identity
```

```
Out[6]: array([ 1.,  1.,  1.,  1.,  1.])
```

Multiply the BMI_array by your identity vector

```
In [7]: #Your code here
        BMI_array * identity
```

```
Out[7]: array([ 24.9613063 ,  21.28692715,  29.02550097,  25.62324316,  30.62382485])
```

Level Up: Using NumPy to Parse a File

The pandas library that we've been using is built on top of NumPy; all columns/series in a Pandas DataFrame are built using NumPy arrays. To get a better idea of a how a built in method like pd.read_csv() works, we'll try and recreate that here!

```
In [8]: #Open a text file (csv files are just plaintext seperated by commas)
        f = open('bp.txt')
        n_rows = len(f.readlines())
        print('The file has {} lines.'.format(n_rows)) #Print number of lines in the file
        f = open('bp.txt') #After using readlines, we must reopen the file
        n_cols = (len(f.readline().split('\t'))) #The file has values seperated by tabs; we read the first line and check it's length.

        f = open('bp.txt')
        #Your code here

        #1) Create a matrix of zeros that is the same size of the file
        matrix = np.zeros([n_rows, n_cols])
        #2) Iterate through the file: "for line in f:"
        for n, line in enumerate(f):
            #3) Update each row of the matrix with the new stream of data
            #Hint: skip the first row (its just column names, not the data.)
            if n > 0:
                matrix[n,:] = line.split('\t')
        #4) Preview your results; you should now have a NumPy matrix with the data from the file
        matrix
```

```
The file has 21 lines.
```

```
Out[8]: array([[   0.  ,    0.  ,    0.  ,    0.  ,    0.  ,    0.  ,    0.  ,
                   0.  ],
               [   1.  ,  105.  ,   47.  ,   85.4 ,    1.75,    5.1 ,   63.  ,
                  33.  ],
               [   2.  ,  115.  ,   49.  ,   94.2 ,    2.1 ,    3.8 ,   70.  ,
                  14.  ],
               [   3.  ,  116.  ,   49.  ,   95.3 ,    1.98,    8.2 ,   72.  ,
                  10.  ],
               [   4.  ,  117.  ,   50.  ,   94.7 ,    2.01,    5.8 ,   73.  ,
                  99.  ],
               [   5.  ,  112.  ,   51.  ,   89.4 ,    1.89,    7.  ,   72.  ,
                  95.  ],
               [   6.  ,  121.  ,   48.  ,   99.5 ,    2.25,    9.3 ,   71.  ,
                  10.  ],
               [   7.  ,  121.  ,   49.  ,   99.8 ,    2.25,    2.5 ,   69.  ,
                  42.  ],
               [   8.  ,  110.  ,   47.  ,   90.9 ,    1.9 ,    6.2 ,   66.  ,
                   8.  ],
               [   9.  ,  110.  ,   49.  ,   89.2 ,    1.83,    7.1 ,   69.  ,
                  62.  ],
               [  10.  ,  114.  ,   48.  ,   92.7 ,    2.07,    5.6 ,   64.  ,
                  35.  ],
               [  11.  ,  114.  ,   47.  ,   94.4 ,    2.07,    5.3 ,   74.  ,
                  90.  ],
               [  12.  ,  115.  ,   49.  ,   94.1 ,    1.98,    5.6 ,   71.  ,
                  21.  ],
               [  13.  ,  114.  ,   50.  ,   91.6 ,    2.05,   10.2 ,   68.  ,
                  47.  ],
               [  14.  ,  106.  ,   45.  ,   87.1 ,    1.92,    5.6 ,   67.  ,
                  80.  ],
               [  15.  ,  125.  ,   52.  ,  101.3 ,    2.19,   10.  ,   76.  ,
                  98.  ],
               [  16.  ,  114.  ,   46.  ,   94.5 ,    1.98,    7.4 ,   69.  ,
                  95.  ],
               [  17.  ,  106.  ,   46.  ,   87.  ,    1.87,    3.6 ,   62.  ,
                  18.  ],
               [  18.  ,  113.  ,   46.  ,   94.5 ,    1.9 ,    4.3 ,   70.  ,
                  12.  ],
               [  19.  ,  110.  ,   48.  ,   90.5 ,    1.88,    9.  ,   71.  ,
                  99.  ],
               [  20.  ,  122.  ,   56.  ,   95.7 ,    2.09,    7.  ,   75.  ,
                  99.  ]])
```

Summary

In this lab, we practiced creating NumPy arrays from both lists and rages. We then practiced peforming math operations like converting imperial measurements to metric measurements on each element of a NumPy array to create new arrays with new values. Finally, we used both of our new NumPy arrays to operate on eachother and create new arrays containing the BMIs from our arrays containing heights and weights.