Pandas!

In this section we'll see our first Python Library. But before we get into that, we need first take a look at how libraries work!

Libraries

A Library in Python is a collection of functions that let you do interesting things without writing raw code to do it.

- You import an entire library in order to use it in your code,
  - You can give it an alias,

    ```
    # Import the pandas library and call it 'pd'
    import pandas as pd
    ```

  - or just use its name

    ```
    # Import the sys library (as sys)
    import sys
    ```

- You can import only the functions you want to use

  ```
  # Import the 'ceil' function from the 'math' library
  from math import ceil
  ```

How you import libraries affects the way you use them in your code.

The `ceil` function in the `math` library lets you find the value of rounding up a number. Let's take a look at some examples:

```
import math as ma
ma.ceil(9.7)
```

```
import math
math.ceil(9.7)
```

```
from math import ceil
ceil(9.7)
```

```
Python comes with some libraries already installed.

The 'math' library is one of them. These libraries that come
pre-installed are called "The Python Standard Library"

One of the main benefits of Python is that there is a large and
well-supported community of handy libraries that you can install
and use for free!
```

The Python Standard Library

Today we will be learning about the pandas library. Pandas lets us play with data!

The pandas library

In your notebook environments we have pre-installed pandas
for you.

(13) - Import pandas into your notebook and give it 'pd' as an alias.

Worked Example: Movie Reviews

Figure 1:

We found some data about movies, these files are located under the folder `data` on your instances.

We can read each of these files into what we call a `DataFrame` and play with the data. This `DataFrame` is an object, and as such, it has many functions we can use to investigate it and have fun!

(14) - Use the read_csv function from pandas to read the file
"data/movies.csv", assign this file to a variable, "movies".

(14.1) - Look at the first 5 rows of your data using the "head"
function on "movies"

```
movies.head(5)
```

What are the columns in this data?

(14.2) Looking at the values, what do you think the "type" of the

```
data in each column is? Write your guesses down.

On Indicies

An index is used by pandas to provide a unique identifier for
each row. If none is specified, Pandas will automatically insert
one based on the row number.

We can set multiple columns to define the index as long as the
combination of values is unique. This is called a multi-index.

Group Question 5 - Can you think of a common multi-index that
you use in your everyday life?
```

We can access attributes of the DataFrame to learn things about it.

For example if I want to know the column names of my dataframe I can do the following:

```
movies.columns
```

```
(15) - Investigate the following attributes:
- shape
- dtypes
```

We can investigate a full column in our DataFrame. the column when referenced by the index, is called a "Series".

Let's investigate our data a bit more closely, and look at the "title" column in our `DataFrame` as follows:

```
movies['title']
```

This results in what Pandas calls a `Series`.

```
On Series and DataFrames

Pandas makes a differentiation between "Series" and "DataFrames".
This becomes important when thinking about what functions we can
perform on them.

Intuitively, we will think of a Series as a single column in a
DataFrame which is still accessible via the index.
```

We can locate data in our DataFrame using the attribute `loc`. To access a particular row, or more precisely, the row indexed by `27`, we would write,

```
movies.loc[27]
```

We can also access a particular set of rows based on a value in the column:

```
movies.loc[movies['title']=="Ghostbusters (2016)"]
```

```
(16) - Read in the file "ratings.csv" in the data folder.

(16.1) - How many ratings are in this file?

(16.2) - Find all the ratings for userId == 14

(16.3) - Asumming 5 is the best score, what is the title of their
 best rated movie?

(16.4) - What are the genres of this movie?

(16.5) - Challenge: Use the "sort_values" function on your ratings
DataFrame and sort them according to the "rating" column. Can you
sort them from largest to smallest, ie. descending
```

Hint: You can find documentation https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.sort_values.html)

Pandas makes it really easy for us to add columns to our DataFrame. Suppose we hate this 5-star rating, instead we want to see a score out of 100, and we want the name of the column to be "score"

```
ratings['score'] = 100 * ratings['rating']/5
```

We can also find out how many unique different "scores" we assigned, by using the "unique" function on the "scores" series.

```
(17) - Find the unique scores by doing the following:
- follow the code above to assing your ratings data a "score"
- access the "score" column as a series, like we did to access
 movie titles (movies['title'])
- use the "unique" function to get all unique scores.
```