**Lab objectives:**

* The purpose of today’s lab is to set up Python, Pandas and Jupyter notebook.
* Refresh your knowledge of Panda’s methods

***Note****: This is not a marked lab.*

1. Make sure Python is installed on your computer. If not you can download the latest version from here: <https://www.python.org/downloads/>

If you need a refresher on Python – this is a good tutorial:

Learn Python [https://www.learnpython.org](https://www.learnpython.org/)

1. Install Jupyter notebook (if Jupyter notebook doesn’t run on your laptop you can use any Python IDE or even text editor)

Jupyter notebooks: [https://jupyter.org](https://jupyter.org/)

Tutorial: <https://realpython.com/jupyter-notebook-introduction/>

Alternatively, you can use Google Colab <https://colab.research.google.com/>

(You don’t have to use Jupyter or Colab, but this is what I’ll be using in class)

1. Install Pandas: [https://pandas.pydata.org](https://pandas.pydata.org/)

Pandas tutorials:

<https://pandas.pydata.org/pandas-docs/stable/getting_started/tutorials.html>

# Getting started with Pandas

1. Create a new Jupyter notebook. Open the dataset using Pandas and print it. The option header=None indicates that our data doesn’t include a header row

A screenshot of a computer

Description automatically generated

1. Each of the “columns” of the tabular data is a Pandas data structure called **Series**, it behaves like a list.

Try accessing all values in a column, or individual values, e.g.

loans[2] will return all ages

loans[2][5] will return 61

You can read more about Series in this tutorial. Try out the examples!

<https://towardsdatascience.com/gaining-a-solid-understanding-of-pandas-series-893fb8f785aa>

1. You can use **df.columns** to set the headings of your dataframe, e.g.

**loans.columns = ['id', 'occupation', 'age', 'loan-to-salary', 'outcome']**

Now you can access the list of all ages but referring to that feature by its name

**loans[‘age’]** or **loans.age**

1. You can check the type of the values in the dataframe by using **df.dtypes.** The type is important as it will determine the kind of operations you may perform on the data.

dtypes returns a **Series** object with the column names as labels and the corresponding data types as values.

You can change the type of the data by using **astype( )** function

1. You can preview the data by using **df.head()** function – that will print the top 5 rows of your data.

You can also ‘slice’ your data and show only specific range (similarly to Python lists and strings), e.g. **df[1:4]** will print all rows from 1 inclusive to 4 not inclusive

# Pandas filtering

Now we’ll try looking at a bigger dataset and try out some conditional filtering

Follow the Kaggle tutorial “**Indexing, selecting, assigning**”:

<https://www.kaggle.com/residentmario/indexing-selecting-assigning>

The dataset used is **winemag-data-130k-v2.csv** (uploaded online, also available at: <https://www.kaggle.com/zynicide/wine-reviews>)

1. After you finish the examples in the tutorial try the following exercises:
2. Identify the list of top wines (wines with score at least 99)
3. Identify the price of the most expensive wine on the list of top wines
4. Print the details (rows of data) for the most expensive top wine(s)

Note: you can save your filtered data in a new dataframe, e.g.

**france\_reviews = reviews.loc[reviews.country == ‘France’]**

will contain only the data rows where the country is France

1. Filter the wines that contain “cherry” in the description.

Hint: you need .**str** accessor from Series