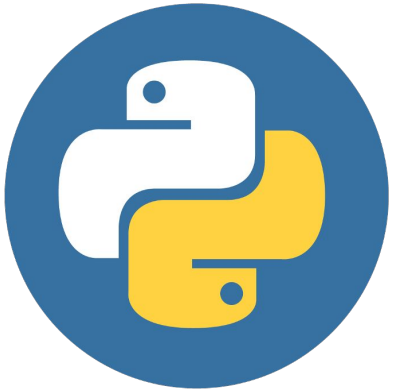
Data Analysis Fundamentals : Python

Global Engineering Challenge - Vaccine Distribution Plan

UNIVERSITY OF TORONTO

Slide by Sarah.L

Data Analysis: Programming Languages



Python

SQL

R

Programming Fundamentals - Variables

- A location where data is stored
- Behave similarly to variables in math - but NOT always numerical
- Each variable has a data type

```
[1]  x = 5
     y = "John"
     print(x)
     print(y)
```

```
5
John
```

```
x = 4       # x is of type int
x = "Sally" # x is now of type str
print(x)
```

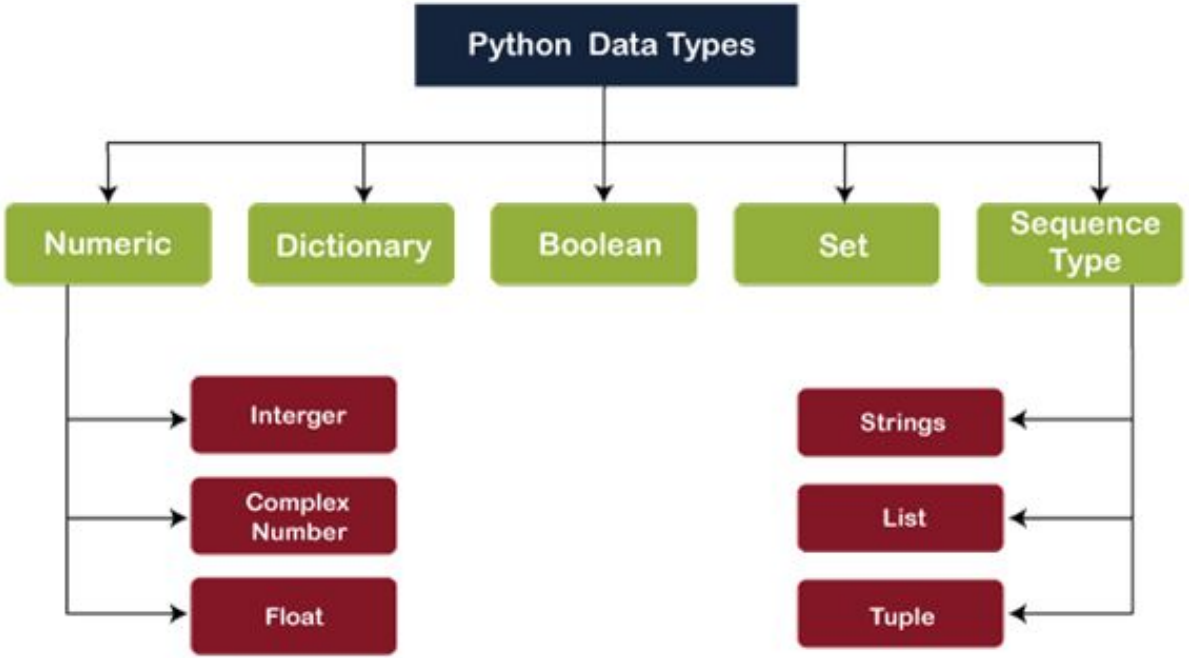
```
Sally
```

Exercise - Variables

Create a variable named `carname` and assign the value `Volvo` to it.

```
        = "      "
```

Python - Data Types



Important ones:

- Integer
- Float
- String
- Boolean
- List

Data Types - int, float

1. Ints = regular integers
2. Floats = decimal numbers
3. Operations:
    a. Addition (+)
    b. Subtraction (-)
    c. Multiplication (*)
    d. Division (/)
    e. Integer Division (//)
    f. Modulus (%)
    g. Exponentiation (**)

```
In [6]: var1 = 6
        var2 = 5

In [7]: var1 + var2
Out[7]: 11

In [8]: var1 - var2
Out[8]: 1

In [9]: var1 * var2
Out[9]: 30

In [10]: var1 / var2
Out[10]: 1.2

In [11]: var1 // var2
Out[11]: 1

In [12]: var1 % var2
Out[12]: 1
```

UNIVERSITY OF TORONTO

Slide by Sarah.L

Exercise - int, float

```
x = 10
y = 8

z = x + y
x = 7
y = 2 ** y

#what are x,y,z?
```

```
a = 5
b = 2

c = a / b
d = a // b

a = b
b = 3

#what are a, b, c, d?
```

Data Types - string (str)

- Strings are groups of alphanumeric characters
- Used widely in text-based applications and printing
- Have a variety of usable “methods”

```
In [1]: str1 = 'Hello World'

In [2]: str2 = 'I am Sarah'

In [3]: str1
Out[3]: 'Hello World'

In [4]: str2
Out[4]: 'I am Sarah'

In [5]: str1 + ' ' + str2
Out[5]: 'Hello World I am Sarah'
```

Data Types - booleans

- Two possible values: **True** or **False**

- In Python:

  False = 0

  True = any other number

  False = empty strings

  True = except empty strings

- Useful in **conditional statements**

```
In [16]: bool1 = 2
         bool(bool1)

Out[16]: True

In [17]: bool2 = 0
         bool(bool2)

Out[17]: False

In [18]: bool3 = "abc"
         bool(bool3)

Out[18]: True

In [19]: bool4 = ""
         bool(bool4)

Out[19]: False

In [20]: not True

Out[20]: False

In [21]: not False

Out[21]: True
```

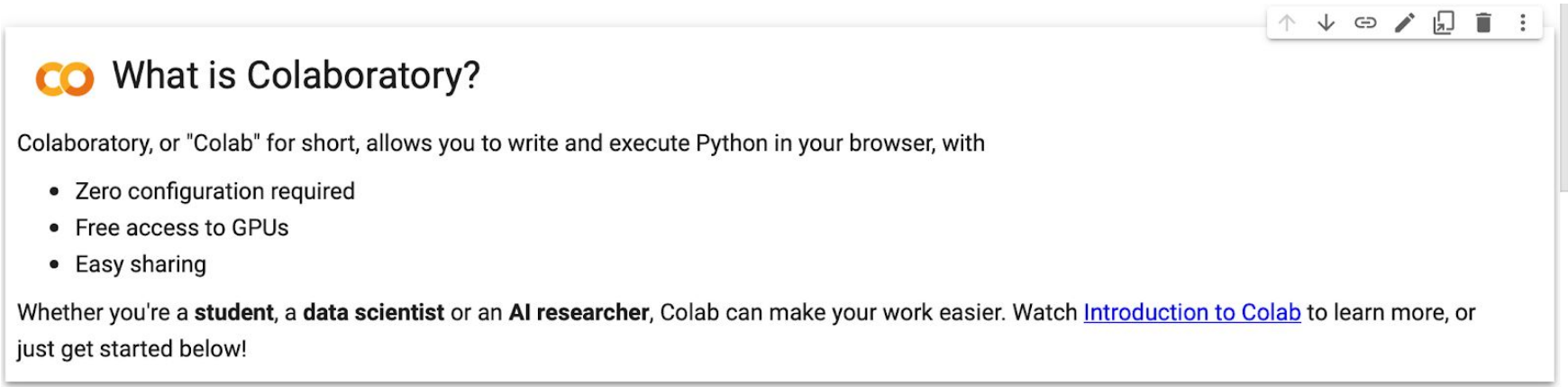
Exercise - booleans

Would each of the following statements be True or False?

```
In [ ]: (4 == 6)          # == means 'equal to'
        2+2 != 7-3        #!= means 'not equal to'
        not 1 == 0
        not not 0 == 0.0
```

Installing Google Colaboratory

https://colab.research.google.com/



What is Colaboratory?

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser, with

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch Introduction to Colab to learn more, or just get started below!

https://www.youtube.com/watch?v=inN8seMm7UI

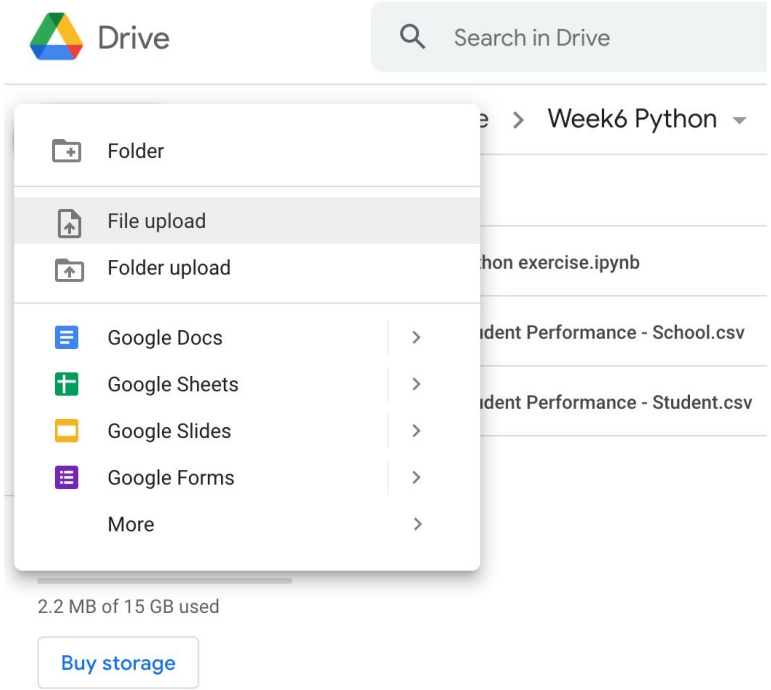
Helpful Resources: Introduction to Colaboratory Video

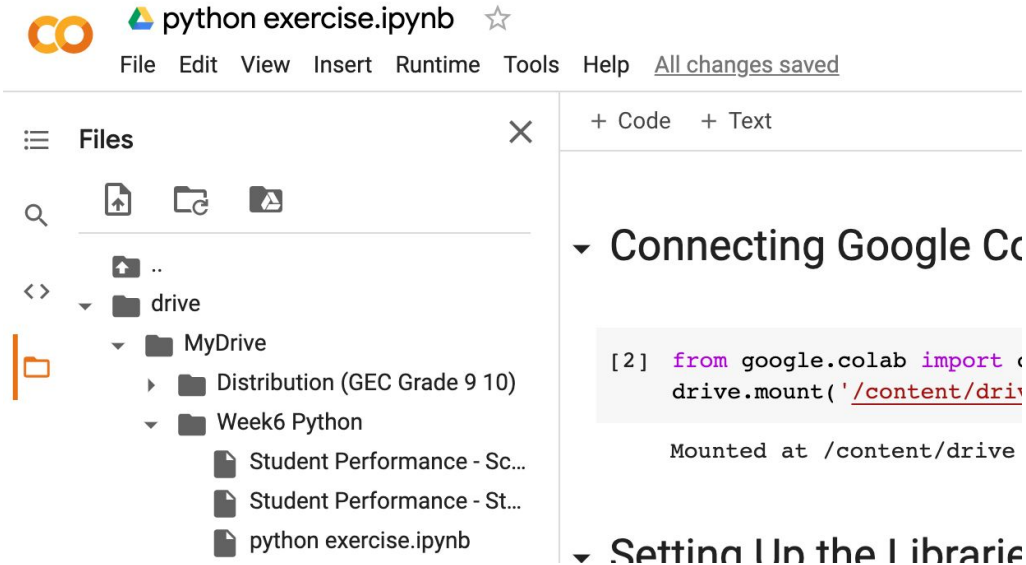
Slide by Sarah.L

Data Visualization in Python

1. Connect Google Colab with Google Drive
2. Import the necessary modules
3. Set up the data to model
4. Decide on the type of model
5. Graph your results

Set up the data to model



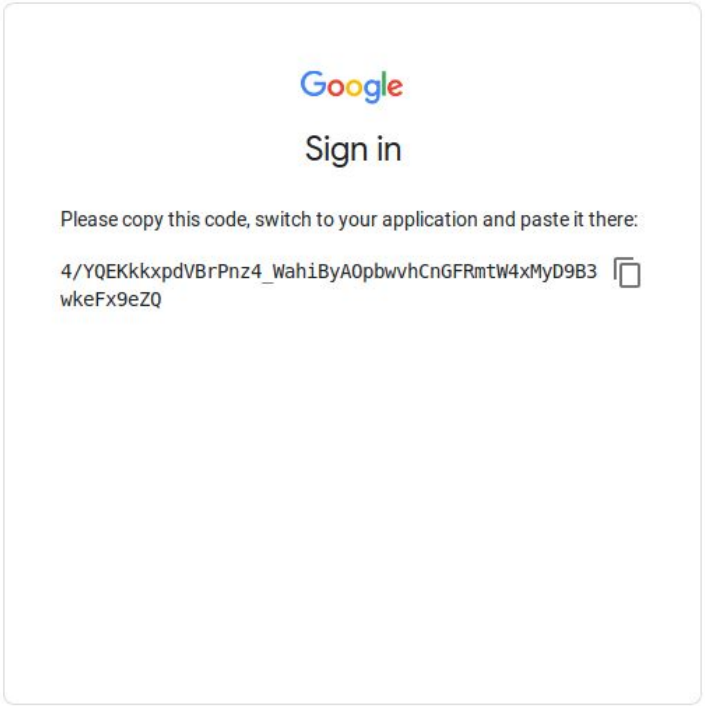


1. Connect Google Colab with Google Drive

▾ Step 1. Connecting Google Colab with Google Drive

```
[2]  from google.colab import drive
     drive.mount('/content/drive')

     Mounted at /content/drive
```



2. Importing the necessary modules

Python has a wide variety of useful libraries containing functions used for data analysis. Using the **import** command allows us to reference these helpful libraries

```
import math
```

```
x = 4
y = math.sqrt(x)
y #to use a function from math library, we need to write math.function()
```

```
2.0
```

Libraries for Data Analysis

Step 2. Setting Up the Libraries for Data Analysis

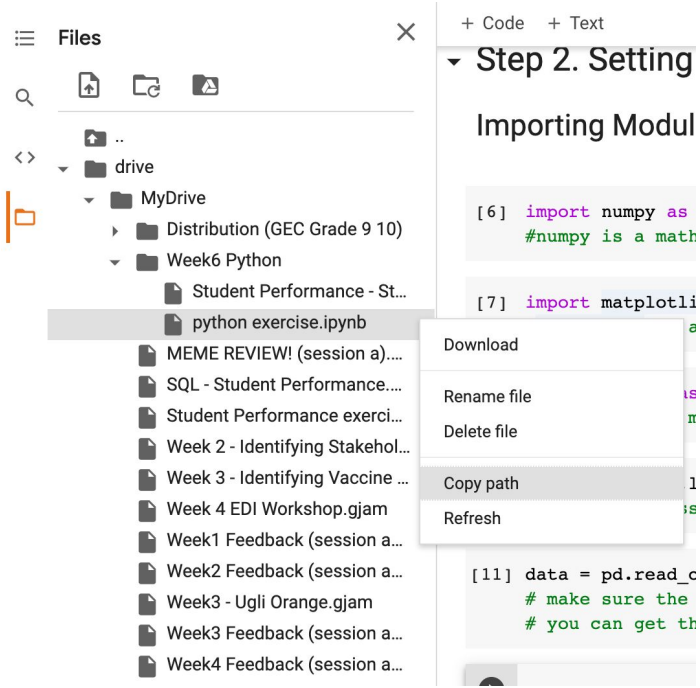
Importing Modules

```
[3]  import numpy as numpy
     #numpy is a mathematics library that contains useful math functions
```

```
[4]  import matplotlib.pyplot as plt
     #matplotlib is a library used for plotting and graphing purposes
```

```
[5]  import pandas as pd
     #pandas is a library used for data manipulation and analysis
```

3. Set up the data to model



Step 3. Setting Up the Data to Model

```
[11] data = pd.read_csv('/content/drive/MyDrive/Week6 Python/Student Performance - Student.csv')
     # make sure the path name is in a string
     # you can get the path link by clicking the three dots next to the file and select'Copy path'
```

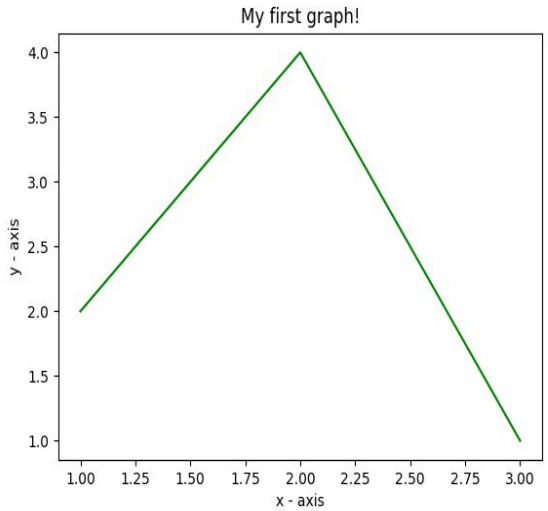
```
[12] data
```

	school_code	school_name	test_prep	math_score	reading_score	writing_score	avg_score	pass_fail	grade
0	GF	Gryffindor	None	72	72	74	73	Pass	B
1	GF	Gryffindor	Completed	69	90	88	82	Pass	A
2	GF	Gryffindor	None	90	95	93	93	Pass	A
3	RC	Ravenclaw	None	47	57	44	49	Fail	F
4	RC	Ravenclaw	None	76	78	75	76	Pass	B
...	...	...	...	...	...	...	...	...	...
995	GF	Gryffindor	Completed	88	99	95	94	Pass	A
996	RC	Ravenclaw	None	62	55	55	57	Pass	D
997	GF	Gryffindor	Completed	59	71	65	65	Pass	C
998	GF	Gryffindor	Completed	68	78	77	74	Pass	B
999	GF	Gryffindor	None	77	86	86	83	Pass	A

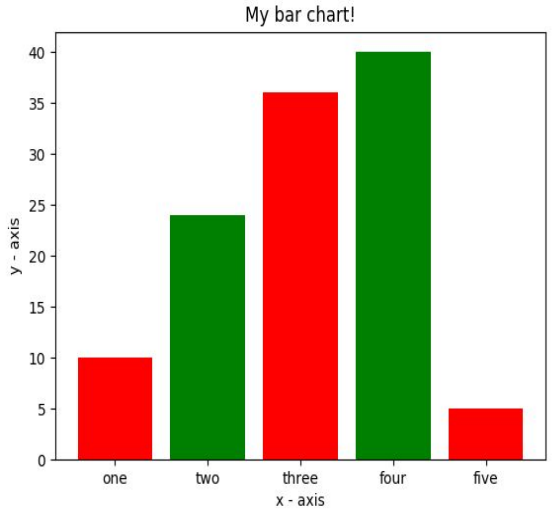
1000 rows × 9 columns

4. Decide on the type of Model

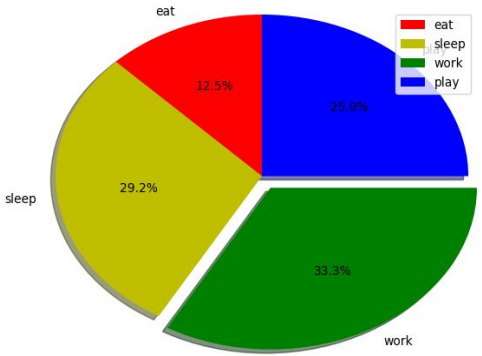
Line Chart



Bar Chart



Pie Chart



UNIVERSITY OF TORONTO

https://www.geeksforgeeks.org/graph-plotting-in-python-set-1/

Slide by Sarah L

4. Decide on the type of Model

Step 4. Decide on the type of Model

1. Line Charts

> A line chart reveals trends or progress over time and can be used to show many different categories of data. You should use it when you chart a continuous data set.

```
#How to create line charts using matplotlib
plt.plot(xAxis,yAxis)
```

2. Bar Charts

> A bar chart is used to show a comparison among different items, or it can show a comparison of items over time. You could use this format to see the revenue per landing page or customers by close date.

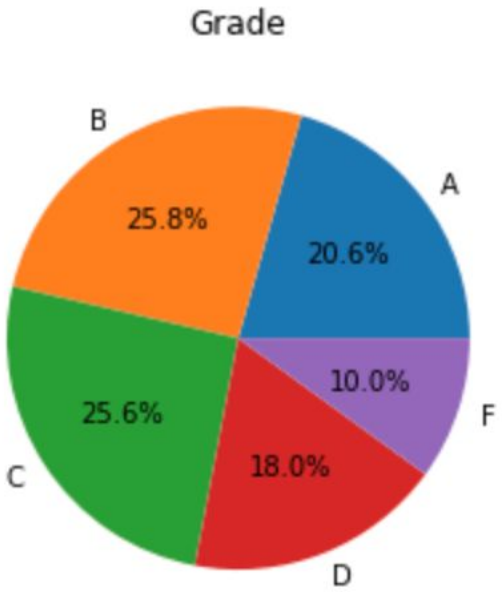
```
#How to create bar charts using matplotlib
plt.bar(xAxis,yAxis)
```

3. Pie Charts

> A pie chart shows a static number and how categories represent part of a whole – the composition of something. A pie chart represents numbers in percentages, and the total sum of all segments needs to equal 100%.

```
#How to create pie charts using matplotlib
plt.pie(my_data,labels=my_labels,autopct='%1.1f%%')
```

5. Visualize Your Results




```
# Creating a Pie Chart

#1. Assign labels for each grade
label = 'A', 'B', 'C', 'D', 'F'

#2. Show the title of a graph as 'Grade'
plt.title('Grade')

#3. Create a pie graph
plt.pie(grades, labels=label, autopct='%1.1f%%')

#4. Show your graph
plt.show()
```