Session 03
Python Foundations

wifi: GA-Guest, yellowpencil

GENERAL ASSEMBLY

Today’s session plan

1800-1810	Standup & Review
1810-1820	Why Python?
1820-1830	Anaconda & Jupyter Notebook
1830-1900	Numerical types, variables, lists, strings, dictionaries
1900-1920	Break
1920-2000	Boolean logic, if statements, while and for loops
2000-2100	Libraries, functions, and classes
Homework: Python, Python, Python	

```
cd ~/Documents/ga-ldn-ds37
git commit -am “your commit message here”
git pull
```

At the end of the session, you will be able to ...

Understand why Python is a powerful language for data science

Apply your fundamental Python skills to perform simple calculations

Launch Anaconda and work with Jupyter Notebook

Import and use library functions

Data Science Part Time

Why Python?

Python Explained

Python is a high-level, open source, object-oriented software programming language often used for scripting, data analysis, and rapid software development

python powered

```
print("Hello, world!")
```

Python Explained

Python is a high-level, open source, object-oriented software programming language often used for scripting, data analysis, and rapid software development

python
powered

```
print("Hello, world!")
```

Group Exercise:

What is programming?

As a class, let's discuss:

- What is programming?
- What's an algorithm?
- What are some examples of algorithms you use in everyday life (non-computer related)?
- What makes a good algorithm?

Python Explained

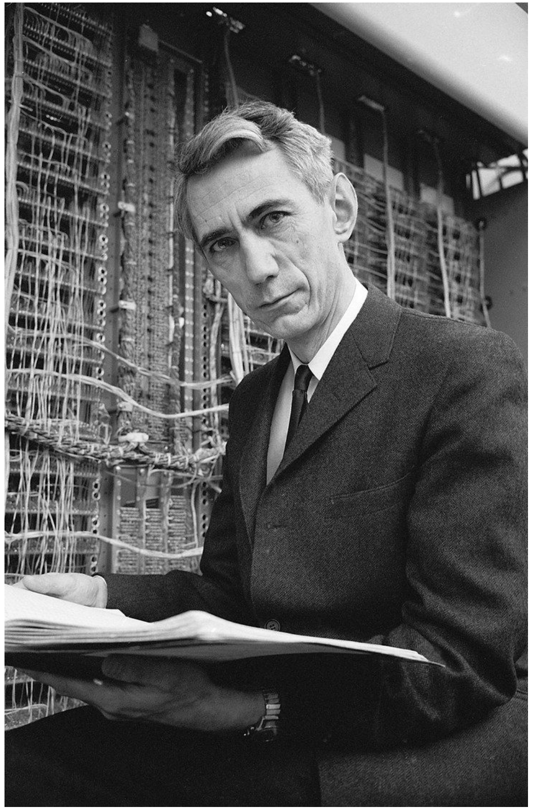
Python is a high-level, open source, object-oriented software programming language often used for scripting, data analysis, and rapid software development

```
print("Hello, world!")
```

Discussion:
The simplest programming language

What’s the simplest, most fundamental language for giving computers instructions?

Binary



- A way of representing information as a string of 1s and 0s
- The most fundamental way of giving instructions to a computer
- All computer processors (the ‘brains’ of a computer) only understand binary

Group Exercise:

Binary

In small groups, discuss:

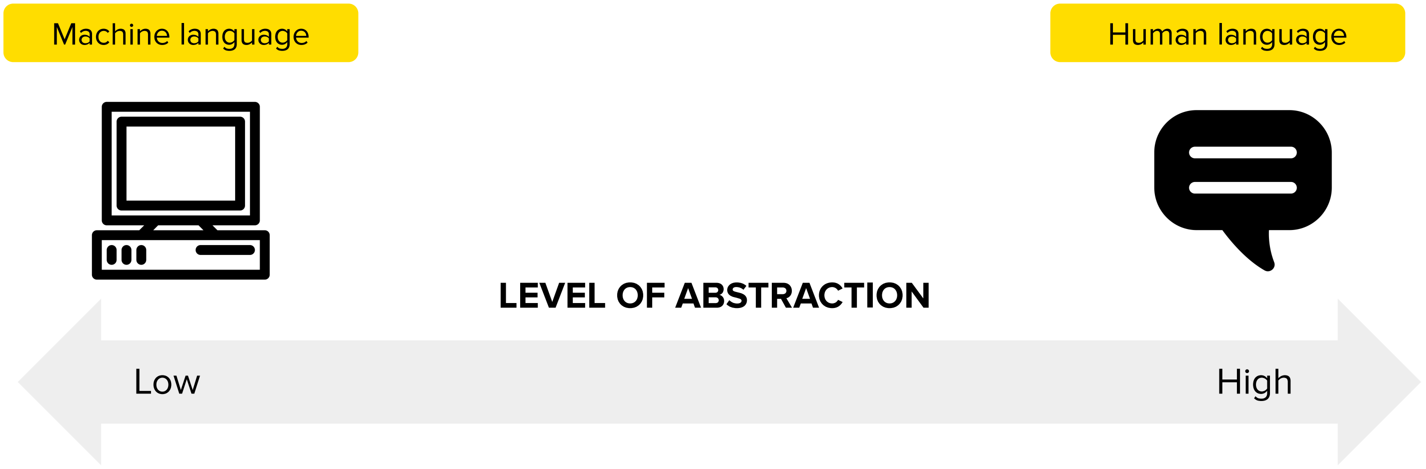
- What are some of the benefits and drawbacks of programming in binary? Would you want to do it?
- If all computers fundamentally only understand binary, how is it that we can write and run code in languages like Python, JavaScript, C++ etc?

Compilers

“It's much easier for most people to write an English statement than it is to use symbols, so I decided data processors ought to be able to write their programs in English, and the computers would translate them into machine code.”

Rear Admiral Grace Hopper (1906-1992)

High-level vs. low-level programming



```
jumpval dd 0 ;
state db 0 ;
```

```
print("hello")
```

Python Explained

Python is a high-level, open source, object-oriented software programming language often used for scripting, data analysis, and rapid software development

python
powered

```
print("Hello, world!")
```

Python Explained

Python is a high-level, open source, object-oriented software programming language often used for scripting, data analysis, and rapid software development

python powered

```
print("Hello, world!")
```

Open Source

Open source code is free for anyone to use, modify, and distribute.

Python is an open source language, so everyone is free to look ‘under the hood’ at its source code.

People can also add to Python’s functionality by writing their own ‘add on’ source code.

This makes Python a powerful, robust and flexible programming language.

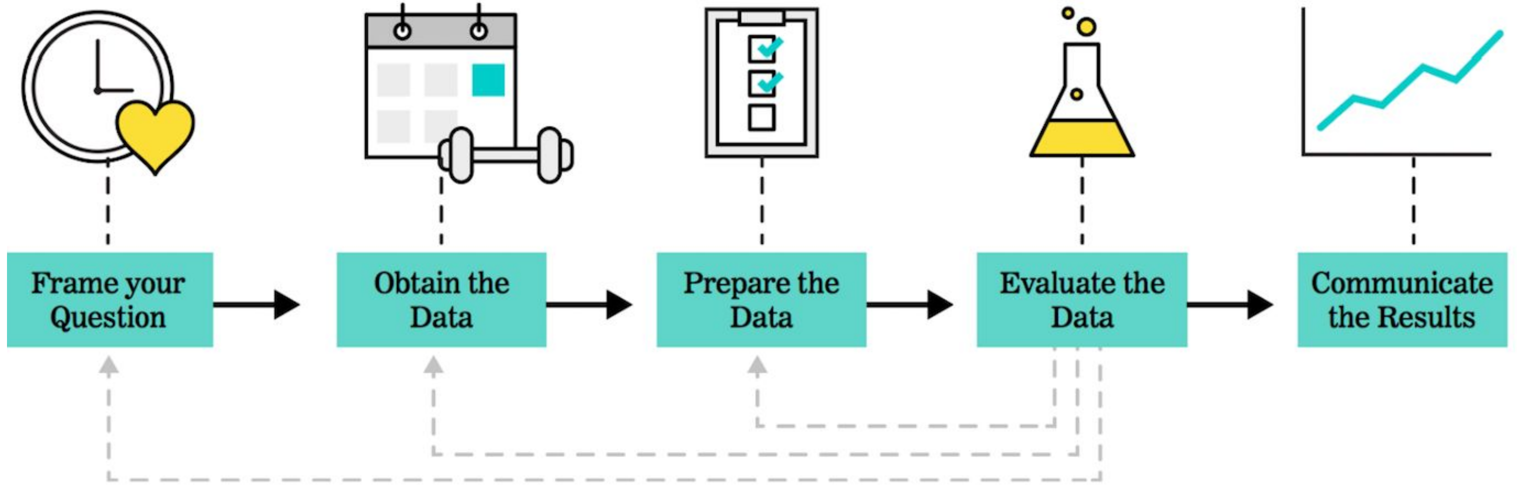
Python Explained

Python is a high-level, open source, object-oriented software programming language often used for scripting, data analysis, and rapid software development

python powered

```
print("Hello, world!")
```

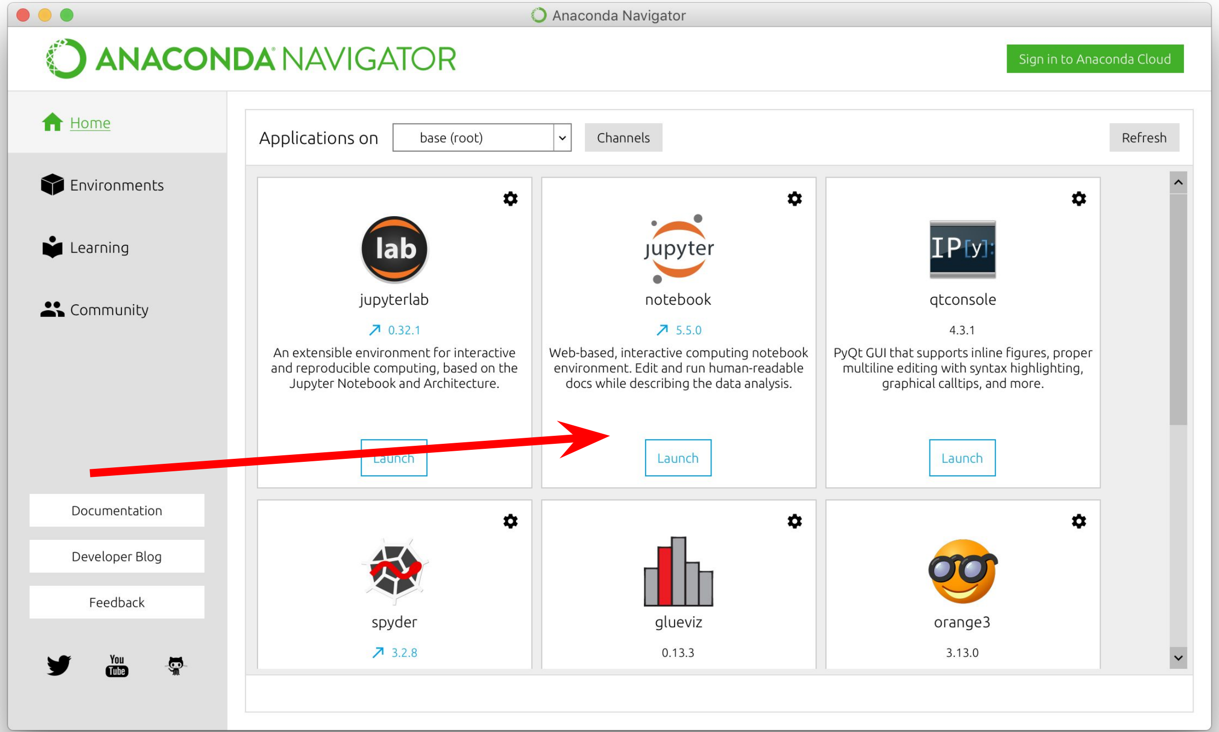
Python is used at every stage of the data science workflow



Data Science Part Time

Introducing Jupyter

Anaconda Navigator



jupyterlab
0.32.1
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

notebook
5.5.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

qtconsole
4.3.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

spyder
3.2.8

glueviz
0.13.3

orange3
3.13.0

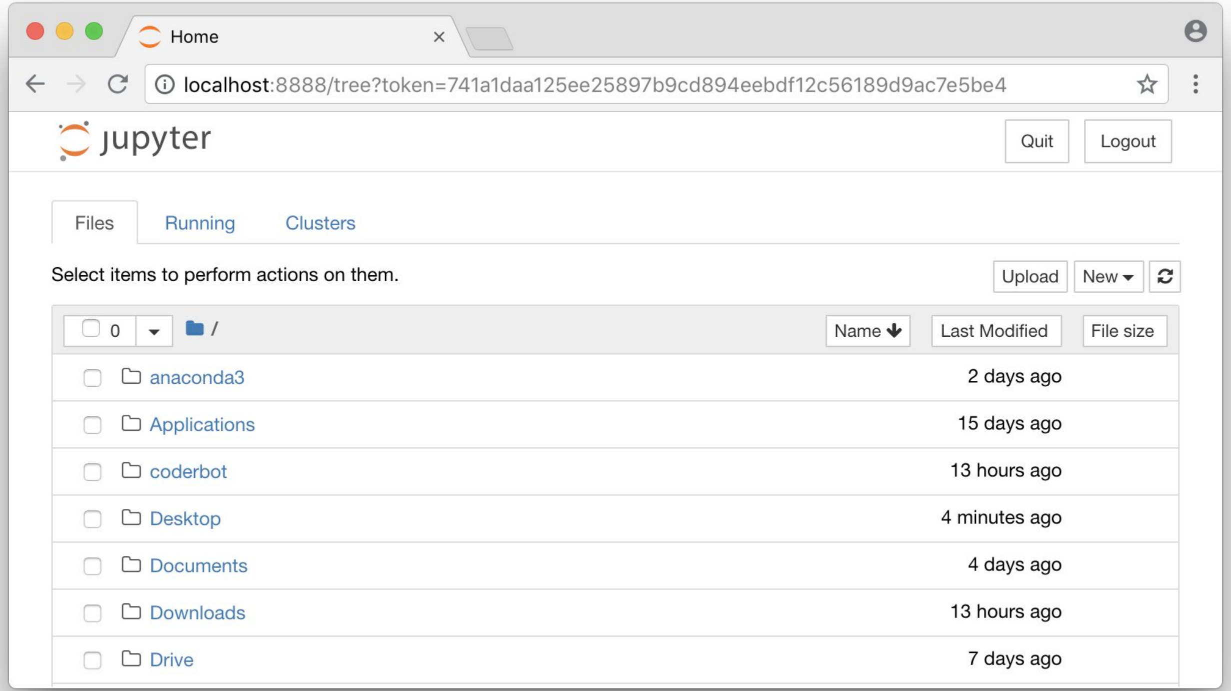
What's Anaconda?

Anaconda is a **distribution** of Python.

This means it's a piece of software that includes Python together with some useful **tools** that make programming in Python easier.

Jupyter Notebook is one of those tools.

Navigate to the location of the newly downloaded folder



Select items to perform actions on them.

Name	Last Modified
anaconda3	2 days ago
Applications	15 days ago
coderbot	13 hours ago
Desktop	4 minutes ago
Documents	4 days ago
Downloads	13 hours ago
Drive	7 days ago

What’s Jupyter Notebook?

Jupyter Notebook (or ‘Jupyter’) is an **environment** for writing and running Python code.

It’s widely used in industry and academia, and has lots of handy features that make it easier to use than many other programming **environments**.

Let’s explore them!

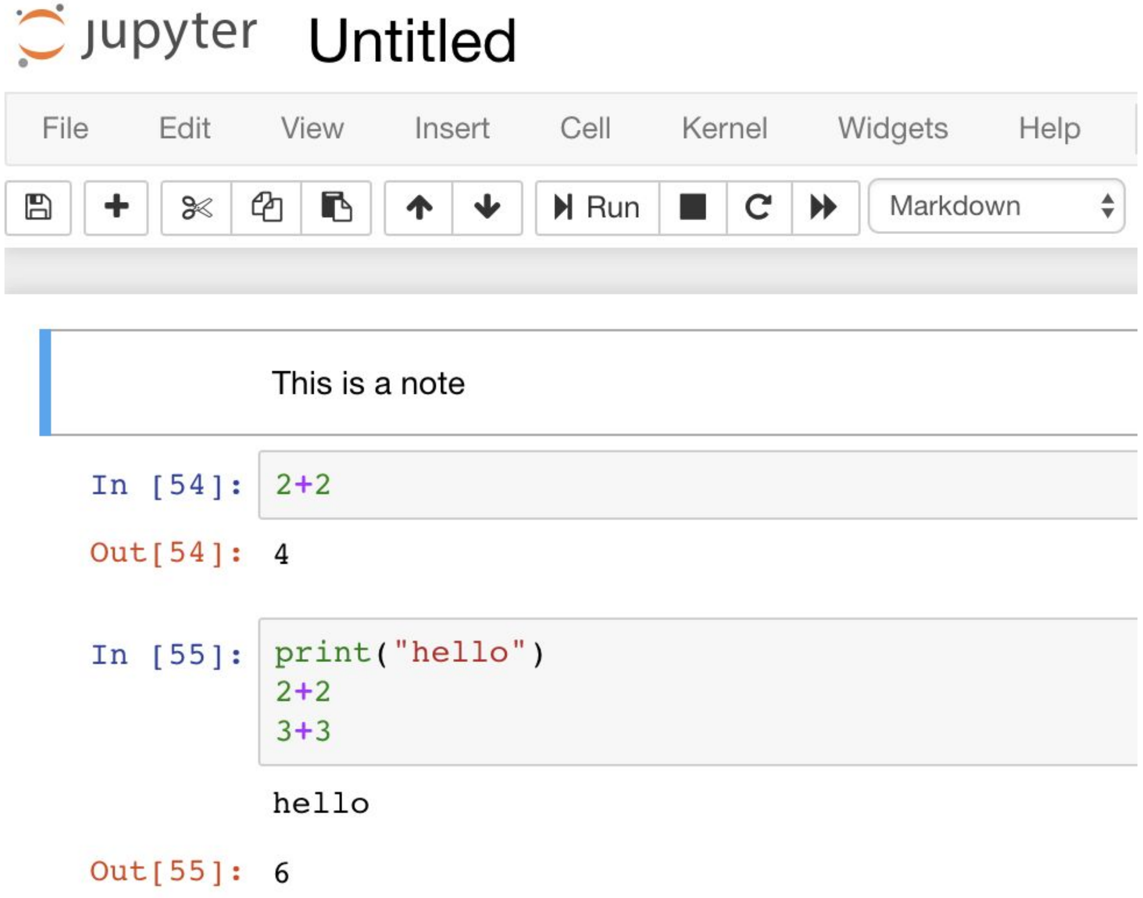
What’s Jupyter Notebook?

Some important points about writing code in Jupyter:

1. Notebooks will **autosave** so don’t panic if you accidentally close a tab
2. You don’t need to be connected to the internet to use Jupyter
3. One cell must **finish** running before another cell can run
4. A code won’t be executed until you run the cell

Jupyter Notebook

- **Cells**
  - **Markdown** for notes
  - **Code** for Python
- **Execution**
  - Shift + return
- **Output**
  - Print (all)
  - Return values (last)



This is a note

```
In [54]: 2+2

Out[54]: 4

In [55]: print("hello")
         2+2
         3+3

         hello

Out[55]: 6
```

Jupyter Shortcuts

Shift+Enter Run cell

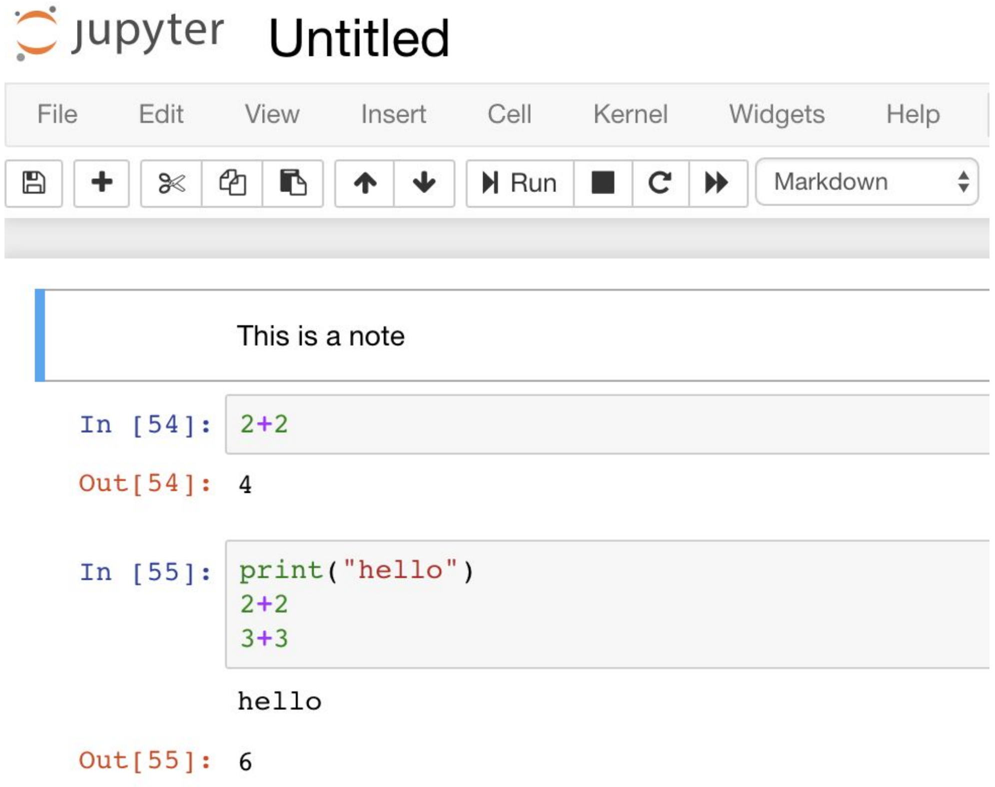
Esc+B Insert cell below

Esc+A Insert cell above

Esc+Y Convert to code cell

Esc+M Convert to markdown cell

Esc+H View all shortcuts



This is a note

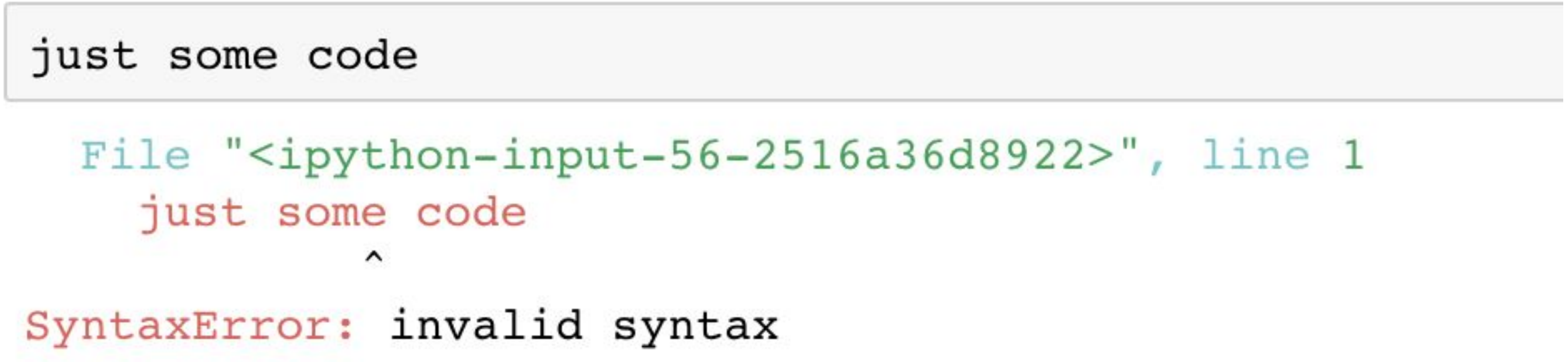
```
In [54]: 2+2
Out[54]: 4

In [55]: print("hello")
         2+2
         3+3

         hello
Out[55]: 6
```

Jupyter Notebook errors

Mistakes happen! Here's what they look like:



```
just some code
```

```
  File "<ipython-input-56-2516a36d8922>", line 1
    just some code
            ^
SyntaxError: invalid syntax
```

1. **Try to understand what went wrong**
2. **Attempt to fix the problem**
3. **Execute the cell again**

Solo Exercise:

Try out Jupyter notebook

Open a new Python 3 Jupyter notebook.

Practise using Jupyter by doing the following:

1. Insert a cell, convert it to a **markdown** cell, insert some text and execute the cell
2. Edit the **markdown** cell so it contains a large heading (hint: use ‘#’ to make a heading)
3. Insert a **code** cell below the **markdown** cell, and execute the calculation ‘2+2’
4. Insert a new cell and delete it immediately

Data Science Part Time

Python Programming Fundamentals

Let’s try Python!

1. For Python, let’s write and run code directly in Jupyter!
2. Open the ‘ds37-03-01.ipynb’ file in Jupyter Notebook
3. Run the code cells in each section to remind yourself of some basic Python syntax
4. Work through the **Now You Try** exercises up to and including the section on **Lists**
5.

Computers Out:

Hello, World!

2 minutes

Let’s get started with our first Python command. It’s useful to think of every Python command in terms of inputs and outputs. Here:

The **input** is the text ‘hello world’

The **output** is the print-out of the text underneath the cell

The **command** is called ‘print’

Discussion:

Types of data

3 minutes

All of these are different **types** of information:

2
2.0
“hello123”
“a”
[3, 4, 5, 6, 7]

Which of these can be...
Divided by 3?
Converted to uppercase?

Different types of data need to be handled in different ways. Python does this by treating different kinds of data as different types.

Discussion:

Types in Python

As a class, let's list as many different Python types as we can, based on our knowledge of the pre-work.

What's special/unique about each type?

Variables

A **variable** is a way of assigning a name to a piece of data.

Variables make it easier and more efficient to store data and perform calculations with it.

Variables can be numbers, strings, lists, etc...

The process of creating a variable is called **declaring** a variable.

Strings

A string is a collection of alphanumeric characters, contained inside quotation marks.

Python treats strings like text.

Different methods can be applied to strings, like converting to lowercase or uppercase.

```
‘hello world’
‘123’
“apple”
```

Lists

A **list** is a sequence of **elements**. We declare a **list** using square brackets and separating each element with a comma.

Each element can be a number, string, or even another list.

```
[1.0, 3, ‘hello’, [1, 2, 3]]
```

Lists can hold multiple types of data.

Elements of a list can be retrieved, or **indexed**, using their position.

Python uses **zero indexing**, so the first element in a list is at position or **index** 0.

Data Science Part Time

Dictionaries

Dictionaries

A **dictionary** consists of **key value pairs** and is an alternative way of storing data.

```
my_dictionary = {name: ‘Maryam’,
fav_food: ‘pizza’,
fav_drink: ‘orange juice’}
```

Instead of retrieving elements from a dictionary using their **position or index** (as with lists) we retrieve elements using their **keys.**

```
my_dictionary[‘fav_food’]
```

Data Science Part Time

Libraries

Libraries

Python has its own built-in functions like **print, len**, **max** etc.

We'll also see later on that we can write our own functions.

But because we want to be smart but lazy, if someone else has written a set of functions that help us perform specialised but common tasks, it's better to use those than to write our own functions from scratch.

This is what a **library** is.

Data Science Part Time

Functions

Functions

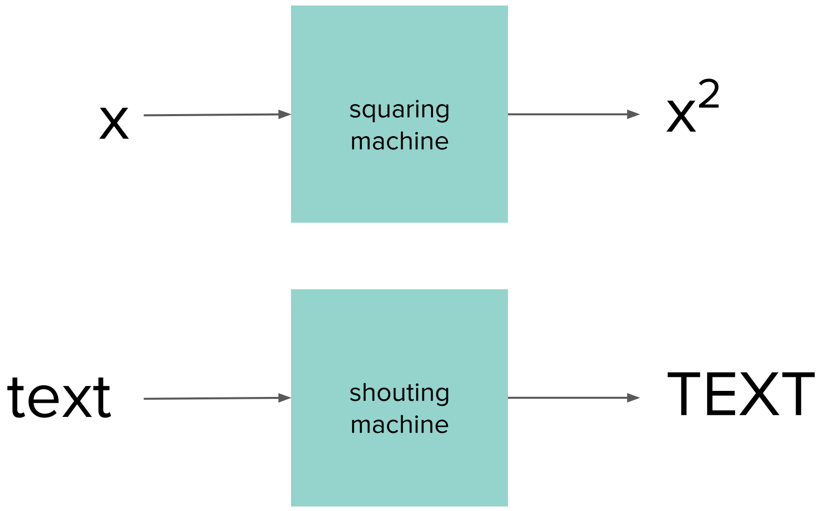
A **function** in Python is like a function in maths.

It takes **inputs**, transforms them in some way, and produces **outputs**.

Python has some **built in functions** that come ‘pre-loaded’ as part of the language, but we’ll often want to write our own functions to perform custom or complex tasks.

Functions

We can think of as a machine that transforms its inputs in a specific way.



Discussion:

Mathematical functions

As a class, let’s list as many Python functions as we can using our knowledge of the pre-work.

What are the inputs and outputs in each case?

Functions

Writing our own functions in Python is a two step process.

1. **Declaring a function**
   This is like building the machine. We specify the name of the machine, how many inputs it takes, how it should transform them, and what outputs it should give us.

2. **Calling the function**
   Once we've built the machine, we can start using it. We give the machine specific inputs, it transforms the inputs and gives us outputs.

Data Science Part Time

Boolean logic

Boolean logic

This is a way of testing whether a statement is **True** or **False**.

A **logical test** will usually involve comparing two or more quantities. Some examples include:

Is one thing larger than another?	`a > b`
Is one thing equal to another?	`a == b`
Is one thing smaller than or equal to another?	`a <= b`

The result of a logical test will always either be **True** or **False**.

The ‘and’ operator

The `and` operator compares **two** logical tests. It takes two inputs.

The result of an `and` operation is **True** if **both** inputs evaluate as **True.**

True `and` True → True

True `and` False → False

False `and` True → False

False `and` False → False

The ‘or’ operator

The `or` operator compares **two** logical tests. It takes two inputs.

The result of an `or` operation is **True** if **either or both** of the inputs evaluate as **True.**

True `or` True → True

True `or` False → True

False `or` True → True

False `or` False → False

The ‘not’ operator

The `not` operator takes one input and inverts it.

`not` True → False

`not` False → True

Intro to Python

Let’s Review

Coming up next session...

- Data acquisition with APIs
- Our first encounter with Pandas
- Project ideas!

Homework

Read through this article and be ready to discuss next session:
https://www.bloomberg.com/news/articles/2019-09-09/jpmorgan-creates-volfefe-index-to-track-trump-tweet-impact

Complete as many of these exercises as you can, up to and including the section on **Python functions**.

https://www.w3schools.com/python/exercise.asp