SECTION 2: PYTHON SETUP / 54 mins, 5 parts

- **3/5 Running Python Code**
  - **-> an outline of ways for running Python code**
    - **-> environments -> the thing you type Python into**
    - **-> ways of working with Python**
      - **-> text editors (you create the .py file in the text editor and then run it in the terminal)**
        - -> these can be customised using plugins and add-ons
        - -> you sacrifice autocomplete etc
        - -> sublime text and atom <- these are the most popular examples
          - -> these are free

      - **-> full IDEs**
        - -> for more different use cases with Python
        - -> these have larger file sizes / add-ins
        - **-> there is a pro version and a community version -> so teams of developers can work on them etc**
        - -> PyCharm and Spyder <- these are the most popular IDEs for Python

      - **-> notebook environments**
        - -> this course starts with notebooks then moves towards .py files
        - -> these use blocks of code (cells) -> and are good for operating in a learning environment
        - **-> the code doesn't need to be ran in the terminal (unlike with text editors)**

        - **-> there is in line mark down, videos, visualisations, images etc are also supported**
          - -> ipynb files (because they have all this extra functionality)
          - -> you can't just double click on these files -> you need to open it in the Anaconda environment (o.e, this is popular in ML / data)

  - **-> using whichever environment for the material in the course depending on what resources are available**

  - **-> how to run Python code**
    - **-> thought process**
      - **-> download the text editor, create a .py file and run it in it in the command line**
      - **-> then take the same code and run it in a JN (Jupyter Notebook)**

    - **-> downloading the text editor**
      - -> in this case it's sublime
        - -> there is a free trial
        - -> unlimited free trial

    - **-> create the .py file using the text editor**
      - **he types out Python**
        - -> print('hello world') #this is standard
        - -> then file save as filename.py
        - -> once the text editor knows which language it's loading -> then it colour codes the text
        - -> when he saves it he also takes note of the directory

- **-> then he opens the file in the terminal**
  - -> **he's cd'd into the location of the file**
  - **-> then -> python filename.py <- you know if it's worked if you click tab and it fills in the rest of the filename**
  - -> then it prints the output of the file in the terminal
  - -> you can also type python -> which will return information about the version of Python etc being used in the terminal
  - -> **to exit out of Python in the terminal -> quit()**

- **-> opening the same file in the Anaconda environment**
  - -> **the difference between the .py file in the txt editor -> and then open it in the terminal approach -> is that that method a) requires you to know linux and b) doesn't include markdown cells like the JN's do**
  - -> he opens the .py file using the Anaconda GUI
    - it's not the same as the .ipynb environment
    - -> so he's opened the .py file in the Anaconda environment, then created an .ipynb environment and pasted it into a code (vs markdown cell)

- **-> so -> either you can create a .py file in a txt editor and then open it in the terminal, or do the entire thing in an Anaconda environment in this example**
- -> **help -> keyboard shortcuts (when you're in an .ipynb in Anaconda)**
- -> in the Anaconda environment -> there is also markdown / code cells
- -> .ipynb
  - -> you can't double click on a file with this extension and expect it to open
  - -> it will open in the default program (in this case sublime, otherwise - error message etc)