SECTION 14: ADVANCED PYTHON MODULES, 2 hours 23 minutes, 13 sections
3/13 Opening and Reading Files and Folders (Python OS Module) - shutil and OS modules

- **-> two builtin modules to open files in Python**
  - **-> the shell utility module (shellutil) and the OS module**
  - -> how you open files in Python and move them around
  - -> **to move / delete files in Python (it's like the terminal in a JN)**
  - -> she's gone into the Anaconda UI and is navigating the file directories

- **-> in the .ipynb file**
  - **pwd <- this is for the file path of the JN**
    - it's reporting back the syntax structure
    - it will show you a \\ or a / in this path depending on which operating system you are using

  - **an example to open and write to a txt file**
    - **f = open('practice.txt', 'w+') <- this opens the txt file (in the same directory as the JN)**
    - **f.write('this is being written into the file')**
    - **f.close() <- this closes the file**

  - **the os module**
    - **import os**
    - **os.getcwd() <- this is the same as pwd, this is get current working directory, this works in any Python script -> but pwd is linux and works in the JN**
    - **os.listdir() <- this is the same as ls in linux (lists out all the files in the current working directory)**
    - **os.listdir('C:\\Users') <- this shows the names of the all the files in the working directory**
    - -> **this is like linux but for Python (navigating file directories etc)**

  - **shellutil**
    - **-> import shutil <- this can be used to move files**
    - **-> sh util <- shell utilities (like bash but for Python)**
    - -> to move a file
      - shutil.move(<- then **shift tab to see which methods (functions are available)**
      - shutil.move(src, dst <- the source and the destination)
      - **shutil.move('practice.txt', 'C:\\Users\\Marcial') <- the file has moved**

  - **OS has a lot of functionality**
    - -> os <- then shift tab, there are three ways to delete the file
    - **-> deleting files in the os module**
      - **<u>os.unlink(path) <- delete the file at the path we provide</u>**
      - **<u>os.rmdir() <- deletes a folder at the path (the folder has to be empty)</u>**
      - **<u>shutil.rmtree(path) <- this removes all files and folders contained in the path -> the methods can't be reversed (they are deleted, they don't go to the trash)</u>**

    - **-> you can send the deleted files to the trash, rather than be deleted straight away -> using import send2trash before using the os module**
      - import send2trash
      - os.list() <- this prints out everything in the current working directory
      - -> then shutil.move(file_path, file_name)
      - -> then os.listdir() <- moved the file back to the directory (this is the same as ls in

linux)

- **-> and then send2trash.send2trash('practice.txt')**
- -> then os.listdir() and the file is no longer in the current working directory

- -> os**.walk <- then shift tab and it's listing out the methods (functions) which can be used**
    - -> the options are top, a directory tree generator
    - **-> for each directory, it's yielding a tuple**

    - **-> for folder, sub_folders, files in os.walk() <- this is tuple unpacking**
        - then there are example file directories
        - -> we have three folders and then in each of them are three folders with txt files in them
        - -> then os.getcwd() <- this is the same as pwd in linux, current working directory
        - -> then she sets that equal to a variable name

        - **-> then she iterates through the file path -> os.walk(file_path)**
            - -> **she's iterating through the different folders in the file path**
            - -> print(f"Currently looking at {folder}") <- this is an f string literal, embedded under the loop which is iterating through the files in the working directory

            - **-> then for f in files:**
                - she is again, this time iterating through the files in the directory (instead of the folders
                - -> she is doing this to print out the files and folders in the directory (in a neater way which isn't ls, in Python not linux)
                - -> printing out the subfolders
                - -> **in other words, you can iterate through all of the folders and files and print out their names to create a tree of the different files and folders (os.walk)**
                - **-> you can also choose ones which were made before a certain date etc**