ONGOING PYTHON QUESTIONS:

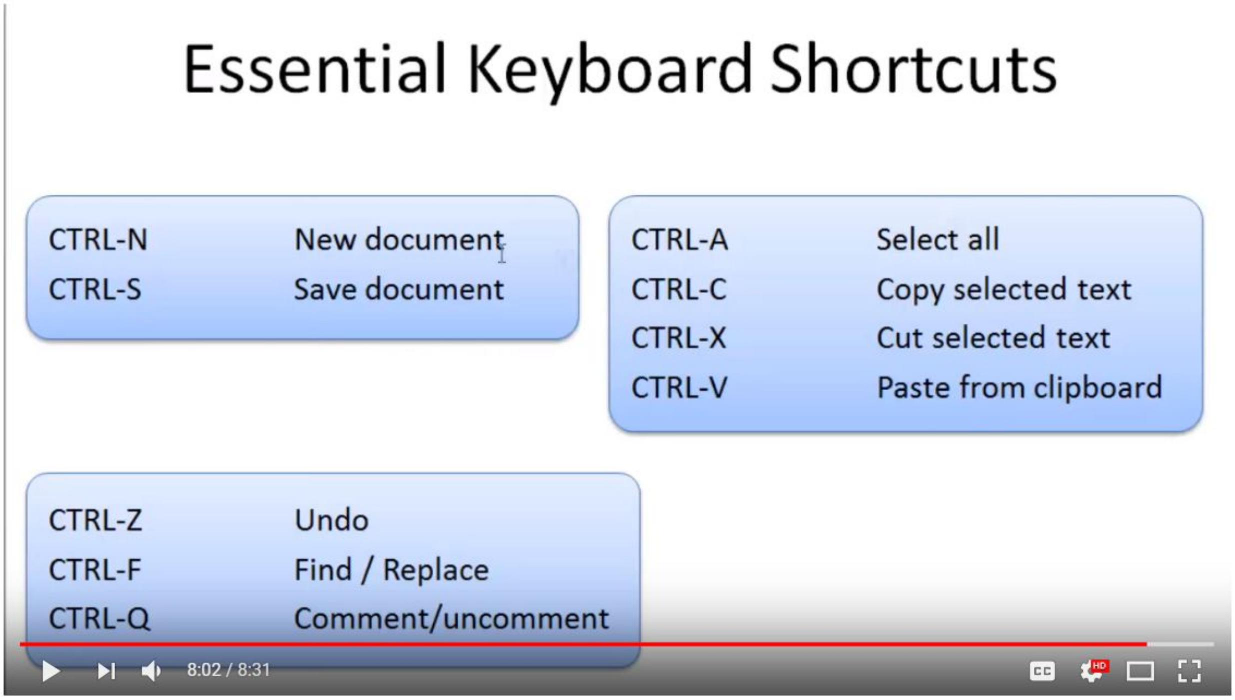
1) While loops: What does while + list mean?
    a) Basically what it means is while that list actually exists.
2) While loop using a flag: does not make sense to me!!
    a) To be safe, just use a flag if you want to make sure that the loop really stops
3) Passing a list to a function - how is the parameter actually set to list? Just a plural?
    a) You can use whatever you want in the parameter. Just make sure you use the **for-in** syntax
4) **CREATING CLASSES -** Storing variables in functions - how does this work? How would you use this function? (for instance, dog.py in Chapter 9) - **DONE**
5) How do **Try-Except** blocks compare with **If-Else** blocks?
    a) You can specify exact errors like "ZeroDivisionError" - and if you KNOW the error, you should specify it
    b) For multiple errors you can use Try-Except-Except loop
    c) Try-Except is checking for errors, If-Else specifying conditions for running your code (but can still be nested within the Try-Except block)
6) Conditional selection - video code didn't work. **DONE**
7) Dataframe indexing

Week 1 Notes

Data path - absolute vs relative

- Absolute paths
    - Windows vs Mac - Windows uses C:\ & Mac uses C:/
    - L through Z are mapped drives; network drives require LPN
    - Libraries are not drives - they are shortcuts
        - Libraries are housed in user folder
    - Many folders are auto-generated by applications
    - Absolute paths start w/root directory
- Relative paths
    - Starts from where you are at; only exists for files or folders within current directory

Week 2 Notes - Variables and Simple Data Types



Essential Keyboard Shortcuts

CTRL-N	New document
CTRL-S	Save document

CTRL-A	Select all
CTRL-C	Copy selected text
CTRL-X	Cut selected text
CTRL-V	Paste from clipboard

CTRL-Z	Undo
CTRL-F	Find / Replace
CTRL-Q	Comment/uncomment

Most helpful tips:

- Text, string, numbers. Remember, you can do things to strings - concatenate, add/remove whitespace, tab etc.
    - Remember that numbers when printed need to be stored in str()!
    - In general, number variables do not need to be in quotations!
- Use \n when you need a line for space (for strings only!)
- int() converts string into integer while str() converts number to string
- input() ALWAYS creates a string
- Variables can store strings + variables; but not variables/text and numbers; everything needs to be converted into same type first!

Week 3 Notes - Lists

Most lists are dynamic - you will build them and then add/remove from the list as your program runs.

- Adding an item - use .append (for iterative list-building) or .insert (to place an item in a specific position)
- Removing an item - use del (if you know the index), remove (if you don't), or .pop() if you want to reuse the removed item. By default, .pop removes the last item in a list, which you will want to store in a variable
- Organizing a list - sort (permanent) vs sorted (temporary); reverse(); len() → helps you to get the indexes of items!

Looping - applying the same action to each item in a list. In this format: for *item* in *list*: *function(item)*

- With function being the action
- Also keep in mind when writing your own `for` loops that you can choose any name you want for the temporary variable that holds each value in the list.
- Don't forget to indent for each new action! Indent = action belongs to the "for" loop
- **For numbers:** use range() to define the range of numbers and list() to automatically convert range into list
- **List comprehension:** easy way to generate a list of calculated numerical outputs
  - Example syntax: `squares = [value**2 for value in range(1,11)]`
- **Tuples vs lists:** Tuples are immutable and they use PARENTHESES!

DON'T FORGET COLON IN YOUR FUNCTIONS

Week 4 Notes - Using Statements with Lists & Dictionaries

Some key keywords for conditional statements:

= vs ==:

= defines a variable

== checks for an inequality

and allows you to include multiple conditions

or allows you to include multiple conditions

in allows you to check if a value is in or not in a list

Combining if statements and for loops allow you to treat special cases within a loop!

You can include multiple actions with each

A dictionary is essentially a series of lists, with information about persons

- Dictionaries are in braces {}
  - Compared with, lists in brackets []
  - Tuples, functions, and number ranges in parentheses ()
- Dictionaries are iterative like lists -> you can build them thru adding/deleting but normally you should just add all key-value pairs at once
- Modifying and adding to a dictionary uses the same syntax (put the new value of the key in brackets)

Looping thru a dictionary

- Use this format: **for** *key, value* **in** *dictionary*.items()
- You can also loop ONLY thru the keys using *dictionary*.keys() or just **for** *key* **in** *dictionary*
- You can retrieve the value of a key by using this syntax in any line of code: *dictionary*[*key*]

Nesting dictionaries

- How do you create a list of multiple dictionaries (about similar items?) You can create them one by one, but oftentimes, it's far more efficient to start with an empty list, then combine a for loop with append. All dictionaries should have similar structure (in terms of the information that is stored)
    - If you loop through a list of dictionaries -
        - Remember that the values in each dictionary must be retrieved through the keys (if you want the values to be printed); and printed thru a "for" loop
- You can also nest a list within a dictionary

Bob's lecture notes:

- Dictionary = hash table
- You can print values of a dictionary by using .values() method
    - Dictionary.get() method also works

Week 5 Notes - User input and while loops

The input function issues a prompt, and then stores the user response in the variable that the input function is being performed on.

While and **For** loops, when used with **If** statements, are collectively known as "control structures" - used in decision branching!

When utilizing loops:

- You might want to start with an empty list or undefined variable
- Adding to variable/counting: use "+" operator after the variable

While loops:

- Make sure your starting condition will actually allow your loop to run (or print the first output) - especially for loops that involve counters/number operations
- Use flags when you need to set multiple conditions for the **while loop**
- Use **while** instead of **for** if you are modifying lists and dictionaries

Functions can be user-defined.

- **Def** *functionname*()
    - Parameters are in the parentheses - parameters specify what information you want to use in the function
    - Arguments are the actual parameter values
        - Positional arguments - make sure arguments follow the same order as parameters
        - Or if you can't/don't want to remember position, use a keyword argument

        - You can set a default value or argument for any parameter, but parameters that have default values must come after parameters that don't have default values
- To make a parameter optional, you need to put an **if** loop in the function's statements
- Parameter can also be a list

Week 6: Classes, Testing Your Code, Exceptions

Classes:

- **Def __init__** is ALWAYS the first method, and it stores the essential information about an instance that you want to use
    - Self is ALWAYS the first parameter, and in the parameter for all subsequent methods, if you want to use the variables that were defined in __init___
    - For subsequent methods, if you want to call any parameters that were defined in __init___ you need to use the exact defined parameter name!

Exceptions:

- But every time your program depends on something external, such as user input, the existence of a file, or the availability of a network connection, there is a possibility of an exception being raised.
- **Try-Except** blocks tells your program how to treat specific errors

Week 7: Data Wrangling using Pandas

How to convert Excel sheet into data frame:
Import **pandas** as **pd**
df = pd.read_csv("*filename***")**
Df
Data frame is essentially just a data table that you can work with in Python.