**Python\_Bootcamp Notes**

# Programs and Software

## VS Code

* Visual Studio Code, also commonly referred to as VS Code, is an integrated development environment developed by Microsoft for Windows, Linux, macOS and web browsers
* Basically it is an interface that allows you to write and edit code in a variety of languages
* Can open Jupyter notebook and use python through it

## GitBash

* At its core, Git is a set of command line utility programs that are designed to execute on a Unix style command-line environment
* Modern operating systems like Linux and macOS both include built-in Unix command line terminals – Windows instead uses the windows command prompt
* Git Bash is an application for Microsoft Windows environments which provides an emulation layer for a Git command line experience

## Python

* Python is an **interpreted**, **high-level**, **general-purpose** programming language.

## Anaconda

* Anaconda is a Python and R distribution software. It aims to provide everything you need for Python “*out of the box*.”

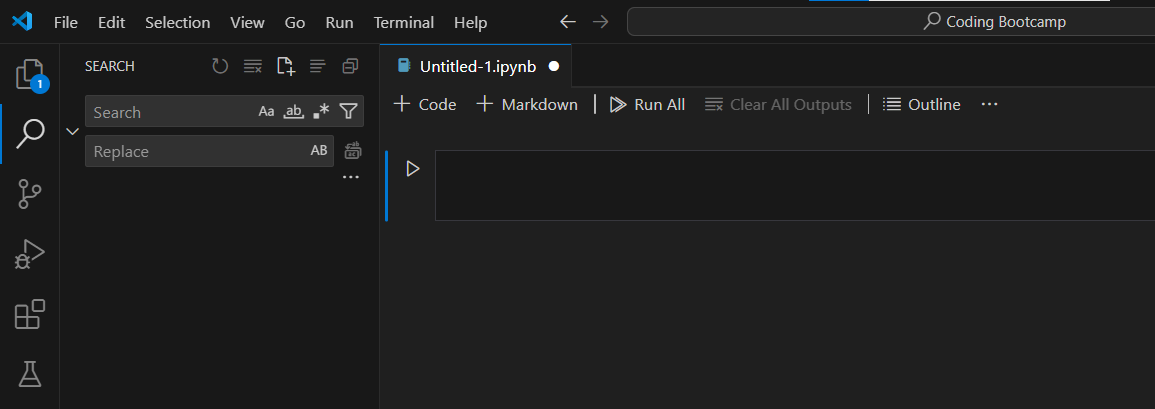
## Jupyter Notebook

* It is an open-source integrated development environment (IDE) that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. For us, it’s essentially our notebook, where we will code along together

# Basics

## Opening a new jupyter notebook

* In VS Code, you can go File > New file
* A prompt will come up to select your file type – choose jupyter notebook

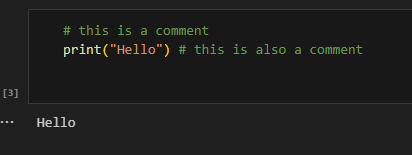


* Code cell: Cells that are executable
* Markdown cell: Text cell – use # to change font size
* Ctrl enter: Executes cell

# Week 2:

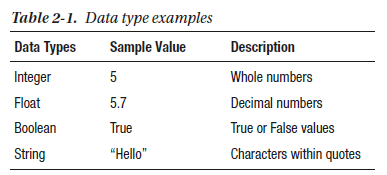
## Comments:

* Comments are like notes that you leave behind, either for yourself or someone else to read.
* In Python, we can write comments using the hash (#) symbol. Any text that follows this symbol will be commented out.



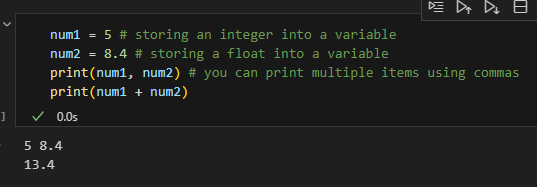
## Data Types:

* Data types are how we define values, likes words or numbers.



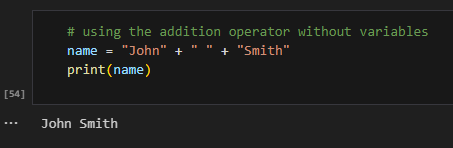
## Variables:

* They allow us to save values into memory using a name that we assign. This lets us use those values later in the program
* We declare a name on the left side of the equals operator (“=”), and on the right side, we assign the value that we want to save to use later.
* Note: Variable names can contain only letters, underscores, and numbers; however, they cannot start with a number.



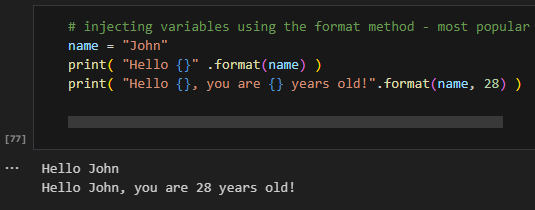
## String concatenation:

* Add one string to the end of another.



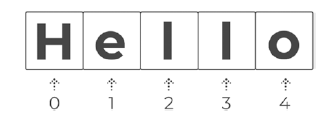
## Formatting Strings (.format()):

* Earlier we created a full name by adding multiple strings together to create a larger string. While this is perfectly fine to use, for larger strings it becomes tough to read
* The format method works by putting a period directly after the ending string quotation, followed by the keyword “format”. Within the parenthesis after the keyword are the variables that will be injected into the string.
* The order of the curly brackets is the same order for the variables within the format parenthesis.
* To include multiple variables in one format string, you simply separate each by a comma



## String Index:

* When a computer saves a string into memory, each character within the string is assigned what we call an “**index.**” An index is essentially a location in memory.
* Note: Indexing in most languages, including python, starts at 0 not 1.
* In order to index a specific element, you use square brackets to the right of the variable name. Within those square brackets, you put the index location you wish to access.



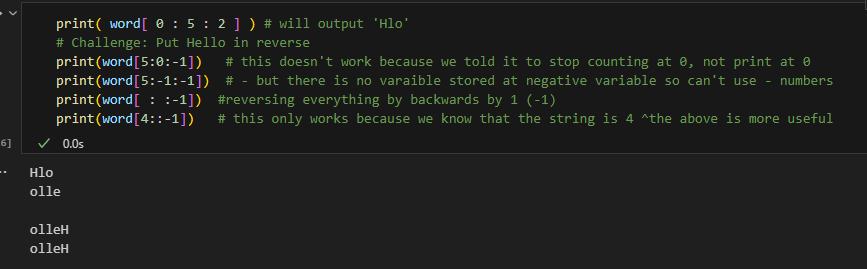
word = "Hello"

print( word[ 0 ] ) # will output 'H'

## String Slicing:

* When only want a piece of the variable

Formatted as *variable\_name[ start : stop : step ]*



## String Manipulation:

.title()

* Often, you’ll run into words that aren’t capitalized that should be usually names. The title method capitalizes all first letters in each word of a string.

.upper() and .lower()

* Converts whole word to capital or lowercase

.replace(“replace this” , “with this”)

* Works like the find and replace tool

.find(“the string we’re searching for”)

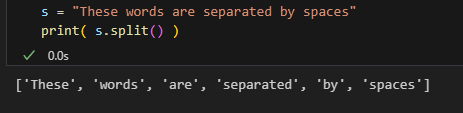
* Output is the starting index of the term

.strip( )

* Default removes spaces (can also .lstrip() and .rstrip()

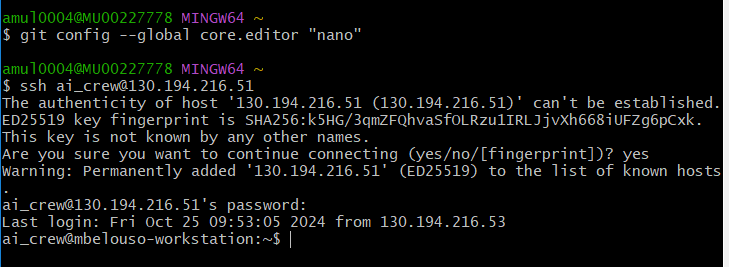
.split()

* Returns a group of words in a sentence stored as a list



# Week 3

## GitBash Crash Course



Bash is programming language – script based language – file browser and file manipulator

But also allows us to execute and run programs – basically everything windows does for us - but condensed into a single command line

Way to interact with every type of computer and can be used to monitor and manage remote computers.

ssh -

username@hostaddress

pwd – print working directory: (shows us where we are at)

~ stands for directory we are in

ls : lists the files in that directory (ie. The folders in that folder)

ls –rtlah : prints everything in the location including hidden files and who owns them and what types they are

drwxrwx-x : from left to right – d= directory, user can read, write and execute, the group can read, write and execute, anyone using the compute can execute

mkdir NAME: make directory and name it

touch name.filetype: ie. touch test.txt – makes a file in git bash

cat name.filetype: displays whats in it (dumps it all)

less test.txt :shows you it line by line as opposed to one big thing – q exits

command >> name.filetype :outputs the command into the file as opposed to displaying it on the screen

^ : means ctrl

nano filename.type : takes you to edit the file – once you’ve finished editing it Crtl O (write out), enter, Ctrl X (exit)

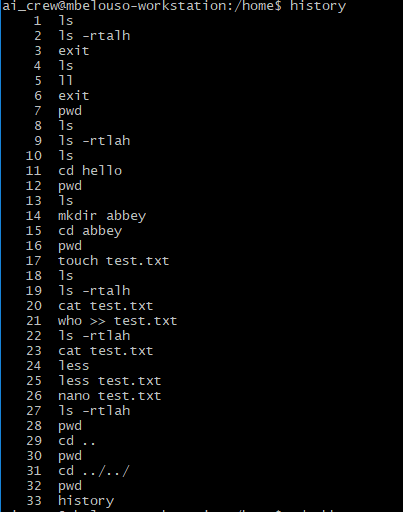
cd . :directory I am currently in

cd .. : takes you back one – can use multiple ../../ if you want to get way back

cd - :takes you back to where you just got out of if you used the ../

exit: gets out of ssh session

mv oldfilename.filetype newfilename.filetype: renames a files



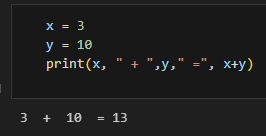
History : shows you all commands you have used

# Command appendix

## Print

print("Your text here")

print(variable 1, variable 2) # will print the defined variables with a space in between



print(“\t{}”.format(your text here) # \t = tab, same as \n = new line

## Type checking

Type(data type here) #will output class ‘data type here’

## .format()

Print(“hello my name is {}” .format(Abbey))

## .title()

Capaitalises first letter of everyword

## .replace(“replace this” , “with this”)

Find and replace tool

## .find(“the string we’re searching for”)

Output is the starting index of the term

## .split()

* Returns a group of words in a sentence stored as a list