**Python introductory workshop Part 1:**

///////////////////////**Resources & Instructions**

**Download the materials from the github repository as you will be coding along throughout the workshop**: https://github.com/resbaz/Intro-To-Python-Master

**Presentation Slides**: <https://mybinder.org/v2/gh/resbaz/Intro-To-Python-Master/master>

**Instruction on how to download, install, and open jupyter notebooks:** <https://github.com/resbaz/Intro_Python_Nov2017/blob/master/Python_Installation.md>

**Google document for sharing our answers to our challenges**: https://docs.google.com/document/d/1jmFPHqiKmBBF-JzDSnQlX-ighTzdpYOmC0xSDip575w/edit?usp=sharing

////////////////////////**Class Notes made by Zhuying Chen, 8th July 2019**

1. Goals: organize our data, and use the commands

2. Aims:

* By learning the syntax around the basic programming structures
* And the jargon that Programmers use to talk about python
* And walk through the foundational programming concepts

3. cells” --- used to run the code, subtype: “markdown cell” used to take notes; use the command “shift+enter” to run the code in cell

In []: #cell used to code

Out []: #output results

4. Specific goals: (1) data type (“ingredients”); (2). Functions (“cooking methods”)

5. Data type:

* Numeric: **integer (int) vs. float**

Python, calculator: +; -; /; \*; \*\* (^2); // (divide and floor); % (modulo—the remainder);

* **String (str)**:
* Using two types of quotes: either double “, or the single ‘

“EOL” in output is the end of line

In []: “3”+’’4’’

Out []: ‘’34’’

In []: “My”+’’ name’’

Out []: ‘’My name’’

* We can also multiply our strings by a number to duplicate it

In []: 'mine '\*5

6. functions:

* **Data type function: type ()** ---tells you the data type that you are working with

In []: type (3)

Out []: int

* **Python length function: len ()**, will tell you how long your string is
* **Data type coercion**: make the data type you want it to be

**Int (), float (),str ()**

In []: int (3.5), round(3.4)

Out []: 3, 4

In []: len(str(3000))

Out []: 4

* **Methods: Specific functions for certain type of object**

Methods are functions specific to a certain type of object: object.method(arguments)

Ex. **string methods**: https://docs.python.org/3/library/stdtypes.html#string-methods

Ex. str.upper(); str.lower(); str.title(); str.capitalize();

* **The Print() statement**

print("Jono", "Garbo","looks","so", "Awesome", "with","a","moustache", "!"\*5)

7. variable

* Variables names for our data; assign the results of an operation into a new variable too
* Python has basically only three rules about naming variables:
* names you define must start with a letter (a-z,A-Z) or underscore (\_) and can be followed by any number of letters, >digits (0-9), or underscores
* names you define cannot be the same as any of Python's reserved words (see below table)
* names are case-sensitive: 'YOU', 'you', 'You', and 'yOu' are all different names in Python
* Note that '-', '+', '\*', and '/' are used by Python for defining operations on data and cannot be used in names
* Try to make your variable names mean something
* Ex.

In []:

team\_name = 'Biofin'

team\_count = '4'

team\_age = '24'

print('Our team name is %s. We have %s members and we are, on average, %s years old' % (team\_name,team\_count,team\_age))

Out []:

Our team name is Biofin. We have 4 members and we are, on average, 24 years old

**/////Summary:**

* basic data types (ints, floats, strings)
* Some functions and methods to manipulate these data types (i.e. type(), len(), str.upper())
* how to store our data in variables and manipulate it

8. **New Data Type: Lists**---[]

* Create the list:

In []: Mylist = list(), tempList = [1,2.5,3,"This", []]

Out []: [], [1, 2.5, 3, 'This', []]

* **Indexing a list**:

***Python actually works based on 0-indexing, meaning that it starts counting from 0 instead of 1.***

This means that if we want to get the first element of our list, we actually have to call list[0], instead of list[1]

Ex.

In []: tempList[1] #the second item in tempList

Out []: 2.5

In []: tempList[-1] #the last item in tempList

Out []: []

In []: tempList[-3] #the last item in tempList

Out []: 3

* **Slicing list, taking more than one item out of a list**

In []:tempList[1:3] #start from second to third item

Out []:2.5,3

In []: tempList[1:] #start from second item to the end item

Out []: [2.5, 3, 'This', []]

In []: tempList[-1:-3] # can not go back

Out []: []

* Mutability and Variable assignment
* **Adding and Deleting List Elements**
* **the .append()**: adds the newest variable onto the end of the list
* **insert()**: put a new item into your list at the location of your choosing
* Deleting items: **del list[index]; list.remove(value); list.pop(index)**

When you want to remove a value but you don't know where it is: list.remove(variable)--- remove() will find and delete the first matching value from your list, and return your modified list back to you.

Ex1.

In []: odds = [1, 3, 4, 5, 4, 7, 9, 9]; odds.remove(4) #remove the first 4 in the list; print(odds)

Out []: [1, 3, 5, 4, 7, 9, 9]

**pop() is index based (remove the value in the specified index in pop**. However, while your list is still modified, pop() actually gives you back the value that you've removed.

Ex2.

In []:

pets = ['dogs', 'cats', 'fish']

print(pets)

pets.pop(0) # Get rid of dogs

dog\_breeds = ['bulldog', 'terrier', 'greyhound']

pets.append(dog\_breeds) # append list of dog breeds available

print(pets)

del dog\_breeds #delete the variable dog\_breeds

pets

**print(pets[-1][-1])** #return the string 'greyhound' from inside this list

Out []:

['dogs', 'cats', 'fish']

['cats', 'fish', ['bulldog', 'terrier', 'greyhound']]

greyhound

* Other list functions: Our first googling exercise: **list.Sort()**

In []:

numbers = [1, 3, 4, 2]

print('Before sorting:',numbers)

numbers.sort() #Sorting list of Integers in ascending

print('After sorting in ascending:',numbers)

numbers.sort(reverse = True) #Sorting list of Integers in descending

print('After sorting in descending:',numbers)

#Sorting list using user defined order

def sortSecond(val): return val[1] #function to return the second element of the two elements passed as the paramater

list1 = [(1, 2), (3, 3), (1, 1)]

print('Before sorting:',list1)

list1.sort(key = sortSecond) # sorts the array in ascending according to second element

print('After sorting in ascending according to second element:',list1)

list1.sort(key = sortSecond, reverse = True) # sorts the array in descending according to second element

print('After sorting in descending according to second element:',list1)

Out []:

Before sorting: [1, 3, 4, 2]

After sorting in ascending: [1, 2, 3, 4]

After sorting in descending: [4, 3, 2, 1]

Before sorting: [(1, 2), (3, 3), (1, 1)]

After sorting in ascending according to second element: [(1, 1), (1, 2), (3, 3)]

After sorting in descending according to second element: [(3, 3), (1, 2), (1, 1)]

9. Summary of learning objects of this workshop:

* Understand the difference between an integer, float and string data types, and storing them as variables
* Creating and storing data in lists, accessing these data via indexing and slicing
* Explaining the difference between functions and methods
* Can recite the three pet peeves of python and how to work around them?