**Lecture 00:**

***How to use the cmd prompt:***

Command Prompt Guide:

*cd*: current directory

*dir:* give all directories

*cd ..*: Go back up through a directory

*cd <*Folder name>: Go to that directory

When entering folder name press tab to autocomplete.

*Cls:* clears the screen

**Terms:**

* **Tuples:** tup Ordered immutable (unchanging) sequence of objects.
  + Tuples are similar to lists.
  + Tuples use parenthesis
* **Dictionary:** elements that can be retrieved through key specifications. Dictionaries have curly braces surrounding them.
* **Sets:** sets don’t have an order to them, sets can only contain one of the same element.
  + Sets cant contain more than one of the same element.
  + Elements can be added to a set using ‘add()’ func.
  + Sets can split a str into a couple of individual elements.
* **Booleans:** Operators that convey true-false statements.
  + True

**Lecture 01:**

Tuples: tup Ordered immutable sequence of objects: (10,”hello”, 200.3)

Python has dynamic typing, which allows for the variable type to change to something else which is different than static typing from other languages like c++

*Commands:*

Type(var) #Gives the type of the variable

Len(“efw”) #Gives the length of a string

Str[3] #gives the 4th character in a string

Str[:3} #grab up to the 3rd character in the string

Str[::4] #go in intervals of 4

Str[3::5] #get letters starting from 3rd character and going to the 5th character in the str

Str[2::] print all letters including the 2nd characters and below.

print('The {} {} {}'.format('fox','brown','quick','woods'))

print('The {f} {b} {q}'.format(f='fox',b = 'brown',q='quick',w='woods'))

print(“The result was {r:1.2f}”.format(r=result))

*Format Strategy* to print number variables within a string

name = “Jose”

print(f’Hello, his name is {name}’)

Strings are not mutable, you can’t use indexing to change individual elements in code

**Lecture02**

New\_list.reverse() # reverses the elements in the new list

New list.sort(reverse = True) #Reverses the elements after they are sorted.

Lists can contain different elements

For example: myList = [4.0, 4, ‘four’] # a valid list variable.

How to get item from nested list? For example get 2 from nested list.

myString = [1,1,[1,2]]

myString[2][1] # gets # 2 out of the string.

Key values

**Examples**

Creating a dictionary and getting a specific element in that dictionary:

d={'key1': ['a', 'b', 'c']}

mylist = d['key1']

mylist[0].upper() # prints “A”

Adding an additional value, which returns “{'k1':100,'k2':200, ‘k3’:300}

d = {'k1':100,'k2':200}

d['k3'] = 300

**Things to remember**

Tuples v. List

Similarities: Lists and tuples have a similar structure

Different: Tuples use parenthesis list uses braces. Tuples provide data integrity, (variables cannot be reassigned contrary to a list.

Lst[‘a’,’b’,’c’] What is the result of lst[1:]?

Starting and INCLUDING b going from b to the end of the list.

To assign values to different variables within a list you can use keys

my\_dict={'k1':123, 'k2':[1,35,56]} #Dictionaries use curly braces

# to access a specific element 56 within the list k2

my\_dict['k2'][2]

#the result returns 56

**Lecture 02**

***Using Jupyter notebook***

**Creating .txt files using Jupyter notebook.**

Type: “%%writefile myfile.txt #*Next line*

<Contents of file> #Go to next line to write to the next line of the file”

**Reading and writing to a file Commands**

myfile = open('myfile.txt') #open a specified texf file.

myfile.read() #read contents of file

*You must then reset the current read position to read the contents again using* myfile.seek0

*Quickly Reading files:*

with open('myfile.txt') as my\_new\_file:

contents = my\_new\_file.read()

*Writing:*

open('myfile.txt', mode = 'a') as myfile:

myfile.write('Goodbye') #prints “Goodbye on the last line

*Different way of writing*

*x = open('test.txt', 'w')*

*x.write("Hello World")*

*x.close()*