Random Python Useful Stuff

## for loop

* for i in range(5): will iterate 0, 1, 2, 3, 4
* Another way to do a nested loop for lists in lists
  + variable = [[1,2],[3,4]]
  + for num1, num2 in variable:

## Populate a list:

my\_list = [I for I in range(1, 51)]

## Combine two lists:

list\_1 = list\_2 + list\_3

## Append:

list\_1 + [989]

## List Repetition:

print(list\_1 \* 4)

## Slice Operator:

: returns a portion of the list. Provide numbers around the slice operator to indicate where you start and stop. *includes the first number but not the second number*.

## Good Practice list of lists:

populous\_cities = [

["USA", "New York City", "Los Angeles", "Chicago"],

["France", "Paris", "Marseille", "Lyon"],

["China", "Shanghai", "Beijing", "Chongqing"],

["India", "Mumbai", "Delhi", "Bangalore"]

]

print(populous\_cities)

## Sorting 2D lists:

numbers = […]

numbers.sort()

for row in numbers:

row.sort()

print(row)

## File(writing mode)

* opening a file to write to it
  + some\_variable = open(“file/path/text.txt”, ‘w’)
    - you can use ‘x’ instead of ‘w’ only difference is that ‘x’ fails if a file of the same name already exists
  + if the file at the end of the “file\_path” does not exist, then a file of that name is created
  + if the file at the end of the “file\_path” does exist, then a file of that name replaces the previous file
* writelines()
  + used to write text to the file
  + some\_variable.writelines(“some text”)
  + can take a list of strings as an argument
    - write does the same thing as writelines() except that it can’t do this
* close a file once your finished
  + some\_variable.close()

## File(append mode)

* opening a file to append it
  + some\_variable = open(“file/path/text.txt”, ‘a’)
  + if the file at the end of the “file\_path” does not exist, then a file of that name is created
  + if the file at the end of the “file\_path” does exist, then that file is appended
* writelines()
  + used to append text to the file
  + some\_variable.writelines(“some text”)
  + can take a list of strings as an arguement
* close a file once your finished
  + some\_variable.close()

File(reading)

* opening a file to read it
  + some\_variable = open(“file/path/text.txt”, ‘r’)
* readlines()
  + returns all text in the file in a list
  + loop through readlines to get individual items
* readline()
  + returns a single line from the text file
  + use with a while loop to loop through file
    - while line != "":
    - print(line)
    - line = read\_file.readline()
  + seek method
    - causes *Python* to go to a specific index character in the text file

## With Open

* combines open and close commands
  + once the end of the indented code is reached, the file is automatically closed
* with open(“file/path/text.txt”, “mode”) as variable\_name:
* can open the file to read and write at the same time
  + with open(“file/path/text.txt”, ‘r’) as variable\_name, open(“file/path/text.txt”, ‘w’) as variable\_name\_two

## Looping through file

* text files end with “”
* while loop with the end of condition “”

## Database

* is a file that is organized for storing data
* module dbm has useful stuff for databases
  + import dbm
  + db = dbm.open(‘captions’, ‘c’)
  + the mode ‘c’ means that the database should be created if it doesn’t already exist, the result is a database object that can be used for most operations
  + create new item in database
    - db[‘random\_key’] = ‘random\_value’
  + access item by calling key

## Tips&Tricks

* use the os module to get functions for working with files, such as returning the current directory
  + csv.reader()
* multiple variable assignment
  + x, y, z = input(‘1’), input(‘2’), input(‘3’)
* printing files
  + print(“text”, file=file\_variable)
* file safety measure
  + if path.exists():
* comprehension
  + short hand way to write a for loop
  + list = for I in variable
* You can define a function inside of another function
  + the nested function can only be called locally by its parent function
* You CANNOT edit a global variable inside a function unless you use the ‘global’ keyword
* .startswith()
* .title()
  + returns a string with the first letter of each word capitalized
* You can use a function object as an argument so that you can call the function inside another function

**ative Assessment 1**

**Formative Assessment**

In physics, force can be calculated with the equation force = mass \* acceleration and acceleration can be defined as acceleration = change in velocity / time. Rearrange the code blocks below to create function declarations for force and acceleration. The acceleration function must be an **internal helper** function for force.