Introduction to Programming (ITAS 185)

# *Lab 5 – Comprehensions and File I/O*

Due: **Friday, October 20 @ 2:00**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will be able to:

* Create a simple Python program using comprehensions
* Create a simple Python program using file i/o

Lab Set-Up

Unzip the folder with the name 185L05\_files and rename it username\_185L05. Username is your logon username that is, first name followed by your last name (mine would be allan.mcdonald).

To Do

Make sure you follow the PEP 8 styling guidelines for all Python code. This includes variable names, function names, etc as well as spacing and code layout.

You are going to create a series of small Python programs each with a specific name. You need to test each program separately and make sure that it works. For some there is a need to accept input from the users and for others there is not.

1. **Comprehensions (to be done in a subfolder of your lab 5 folder named part\_a)**
2. Create a python program called comp\_play.py. Create each of the following sequence datatypes using a list comprehension and then display the result (using a print statement):
   1. Create a list from the elements of a range from 1200 to 2000 with steps of 130, using list comprehension.
   2. Create a new list of the values in the list check\_list = [44, 54, 64, 74, 104] with 6 added to each list item (result [50, 60, 70, 80, 110].
   3. Create a list of all the numbers from 1-1000 that are evenly divisible by 7 (use modulus).
   4. Construct a list from the squares of each element in the list my\_list = [2, 4, 6, 8, 10, 12, 5, 14], if the square is greater than 50 (result [4, 16, 36, 25]).
   5. Create a list of all the numbers from 1-300 (inclusive) that have a 3 in them (not divisible by 3…have a 3 in them like 233 or 37). You will need to convert the integer to a string and then use the find method to find out if the value you are checking has a 3 in it; find returns -1 if there is no matching value and the index of that value if there is a match.
   6. Create a list of tuples consisting of only the matching numbers in these lists list\_a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11] and list\_b = [2, 7, 1, 12]. Result would look like [(1, 1), (2, 2), (7, 7)].
   7. Create a list of all the words in the string “This is a long string with many words even with some that are less than five letters long” that are less than or equal to 4 letters long. You will need to use the split command to break the string into a list of words and then go through that list of words checking the len of each word.
   8. Use a nested list comprehension to find all of the numbers from 1-100 that are divisible by any single digit besides 1 (2-9). You may want to do this using a nested loop first and then translate it to a comprehension.
3. **Files (to be done in a subfolder of your lab 5 folder named part\_b)**
4. Create a file called do\_sum.py. Prompt the user for a filename. Write a function, sum\_column(filename). It is passed the filename input and reads the file input (I would use the readlines method). Each line in the file has exactly one integer. The function returns the sum of these integers. Print out the sum. Sample file sum\_column.txt sums to 40. You can NOT use lambda functions to shorten the array summation. You should add your own files to test.

Hint: You will read strings from the file and need to convert them to integers before adding them together (this should be done with a comprehension).

Hint: Remember, when you read a line from the file using readlines the newline character is included. You must use the strip() method to remove those newline characters.

1. Create a file called count\_letters.py. Open the file letters.txt and read the file contents. Count the number of vowels (aeiou) in the contents of the file (this can be done using a comprehension). Also count all the consonants in the file (bcdfghjklmnpqrstvwxyz) (a different comprehension or a more complex single one). Ignore any other characters. There are 145 vowels and 254 consonants.

Hint: Functions are your friends.

Hint: You should use the read method this time and then you only have one string to go through.

Hint: You can create a list of all the vowels (and consonants). The length of that list is the number of consonants or vowels in the file.

Hint: You will have to convert the string to either upper or lower case to test it as the strings you read will have both upper and lower case letters.

1. Create a file called users\_choice.py that prompts the user for a letter. Once the letter is entered, open and read the contents of the file count\_letters.py. Count the number of times the letter entered is found in the file. Once again, this must be case insensitive so you will need to convert the letters to upper or lower case (your choice). You must use comprehensions.

In the test file there are 20 y’s; 14 z’s and 15 q’s.

**To submit**

When you have completed the lab exercise, call me over and we’ll go over it together. Then, create a single zip file called username185L05.zip and copy the file to the Moodle page for the course.