CSCE 1035 Lab 11 Fall 2021

1. **zyBooks Labs**

   Please follow the link on Canvas to complete the following zyBooks labs:

   - 8.16 *LAB: Varied amount of input data*
   - 8.17 *LAB: Filter and sort a list*
   - 8.18 *LAB: Middle item*
   - 8.19 *LAB: Elements in a range*
   - 8.20 *LAB: Word frequencies*
   - 8.21 *LAB: Contact list*
   - *8.23 LAB: Warm up: People's weights (Lists)*

2. **Working with Nested Lists**

   In this lab component you will write a small, but complete Python 3 program called **`Lab11A.py`** that creates and prints a nested list of integers and then use list comprehension to multiply each element in the nested list by two. This can be accomplished as follows:

   a. Create a user-defined function `print_list(list_to_print)` that accepts a list of integers as its argument. This function does not return anything.

      - Use a `for` loop to iterate through `range(len(list_to_print)`. Inside the loop, use the format specifier with `width = 2` to print the list on a single line, with each integer separated by a single space.
      - Outside of the loop, use a single `print()` statement to terminate the line.

   b. Create a user-defined function `print_2D_list(list_2D_to_print)` that accepts a nested list of integers as its argument. This function does not return anything.

      - Use a `for` loop to iterate through `range(len(list__2D_to_print)`. Inside the loop, call the `print_list()` function, passing a row (i.e., a 1-D list) of the `list_2D_to_print` list. This means that you will index `list_2D_to_print` with a subscript to access the row.

   c. In the main part of the program, initialize an integer variable `v` to 0 and create an empty list called `two_d`.

   d. Use a `for` loop to iterate through `range(5)`.

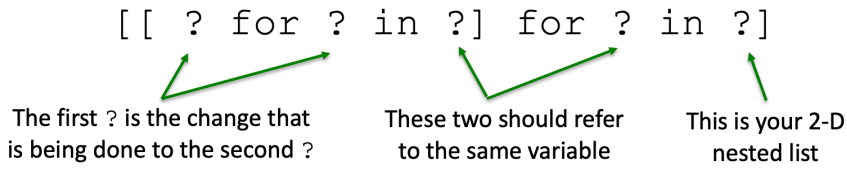
      - Inside this `for` loop, append an empty list your `two_d` list.
      - Use another `for` loop to iterate through `range(4)`.
        - Inside this nested loop, append your integer variable `v` to the current row of the `two_d` list (Hint: Use the control variable for the outer for loop!).

- Increment the integer variable `v` by 1.

e. Call the `print_2D_list()` function, passing the nested list `two_d`, which should now have 5 rows with 4 columns. You may wish to add a single `print()` statement to add a newline character to the terminal.

f. Now, use a nested list comprehension to multiply each integer in `two_d` by 2 and assign the result to a new nested list called `new_2d`.

- You can use this structure to help you in constructing your list comprehension:



g. Finally, call the `print_2D_list()` function again, passing the nested list `new_2d`.

For example, the output should look like this (input shown in **bold**):

```
$ python3 Lab11A.py
 0  1  2  3
 4  5  6  7
 8  9 10 11
12 13 14 15
16 17 18 19

 0  2  4  6
 8 10 12 14
16 18 20 22
24 26 28 30
32 34 36 38
```

Note that you will submit this file to Canvas.

3. Working with Dictionaries

In this lab component, you will count the frequency of specific words in a given sentence, using a variety of techniques while working with dictionaries.

Write a small, but complete Python 3 program called **`Lab11B.py`** that does the following:

a. Create a user-defined function called `add_to_dict(wordlist)` that accepts a list of words (i.e., strings) as its only parameter and returns a dictionary of word-frequency pairs.

- Create a simple dictionary comprehension as follows:

```
wordfreq = [wordlist.count(p) for p in wordlist]
```

This creates a list of word frequencies (e.g., `[1, 1, 2, 1]`) based on the words in the list of words passed to the function.

- We will now use the `zip()` function to match the first word of the word list with the first number in the frequency list, the second word and second frequency, and so on to end up with a list of word and frequency pairs. We will also use the `list()` and `dict()` functions to create a list and dictionary of the word and frequency pairs, respectively, as follows:

```
return dict(list(zip(wordlist, wordfreq)))
```

b. In the main part of the program, prompt the user for and read in a sentence (that may include several words separated by spaces) as a string.

c. Use the `split()` function to convert the sentence string into a list of words.

d. Create an empty list called `wordfreq` to store word frequencies.

e. Iterate through the list of words.

- Use the `count()` function for the list of words for each word in the word list and append that result to `wordfreq`. This should create a list of word frequencies (similar to what we did above with the dictionary comprehension).

f. Print the list of words.

g. Print the list of word frequencies.

h. Now, create an empty dictionary.

i. Call the `add_to_dict()` function, passing in the list of words, assigning the result to the currently empty dictionary just created.

j. Iterate through the dictionary using the `items()` function for the key and value of the dictionary.

- Print each key and value that represent the word and frequency pairs from the dictionary.

For example, the output might look like this (input shown in **bold**):

```
$ python3 Lab11B.py
Enter a sentence: the car will park next to the park in the
street
Word list: ['the', 'car', 'will', 'park', 'next', 'to', 'the',
'park', 'in', 'the', 'street']
Word freq: [3, 1, 1, 2, 1, 1, 3, 2, 1, 3, 1]
the      : 3
car      : 1
will     : 1
park     : 2
next     : 1
to       : 1
in       : 1
street   : 1
```

Note that you will submit this file to Canvas.

Now that you have completed this lab, it's time to turn in your results. Once you've moved the files to your windows machine (using **WinSCP**), you may use the browser to submit them to Canvas for the **Lab 11** dropbox.

You should submit the following files:

- **`Lab11A.py`**
- **`Lab11B.py`**
- **(Note that the zyBooks labs are submitted separately through Canvas.)**

Ask your TA to check your results before submission.

Now that you've finished the lab, use any additional time to practice writing simple programs out of the textbook, lectures, or even ones you come up with on your own to gain some more experience.