CS-521 Homework Assignment 4

Assignment Directions

The specific instructions for these programming exercises are adapted from problems shown in the textbook. Make sure to follow the specific instructions in THIS document.

Each of the 6 programs is worth 10 points for a total of 60 points (100%).

Style Requirements

For all assignments, follow the guidelines in the PEP8 Standards and Best Practices that have been shared to date, along with course specific requirements. The following will be minor deductions:

- Improper naming of programs or zip container
    - Among other things, names for regular programs and zip file must be all lower case.
- Missing program docstring
- Inadequate # line comments (just a few in each program, don't go crazy)
- Going significantly over 80 characters for code/comment lines
- Not naming objects appropriately
    - variables and functions must use snake_case (lower case plus underscores)
    - CONSTANT variables in all upper case
    - Class Objects, which are taught in module 6, as CamelCase
- Asking for input() without descriptive prompts telling the user what is expected.
    - Especially not mentioning the delimiter when input will be split()
- Printing output that is not clearly explained (where necessary)

CS-521 Homework Assignment 4

Chapter 7 Exercises

4.7.1 Given a constant list of integers in the range 1 to 10 inclusive, **use list comprehension** (no explicit loops) to:

- find the sum of the even integers in list L.
- find the sum of the odd integers in list L.

Example of Output:

```
Evaluating the numbers in: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
Even: 30
Odd: 25
```

4.7.2 Given a constant list of integers, write Python code to generate a new list with same number of elements as the original list such that each integer in the new list is the sum of its nearest neighbors and itself from the original list. Print both lists with descriptions. Your code should be able to work with an integer list of any size.

Example of Output:

```
Input List: [10, 20, 30, 40, 50]
Result List: [30, 60, 90, 120, 90]
```

CS-521 Homework Assignment 4

Chapter 9 Exercises

4.9.3 Start with 2 constant lists. One with first names and another of last names.

First validate that both lists are the same size and if not, exit with an error message.

Use zip to create a dictionary with the keys as the last names and the values as the first names. Print the generated dictionary with an appropriate description.

Example of Output:

```
First Names: ['Jane', 'John', 'Jack']
Last Names: ['Doe', 'Deer', 'Black']
Name Dictionary: {'Doe': 'Jane', 'Deer': 'John', 'Black': 'Jack'}
```

4.9.4: Using my_dict = {a':15, 'c':18, 'b':20}, write a program to:

a. print all the keys.
b. print all the values.
c. print all the keys and values pairs.
d. print all the keys and values pairs in ascending order of key.
e. print all the keys and values pairs in ascending order of value.

Example of Output (showing alternative ways **you might choose** to print lists and dictionaries):

```
a. Keys: ['a', 'c', 'b']
b. Values: 15, 18, 20
c. Key value pairs: a: 15, c: 18, b: 20
d. Key value pairs ordered by key: [('a', 15), ('b', 20), ('c', 18)]
e. Key value pairs ordered by value: a: 15, c: 18, b: 20
```

CS-521 Homework Assignment 4

4.9.5: Write the following python program.

Create 3 functions with docstring:

1. letter_counts() takes a string as its argument and returns a **dictionary of the letters** as keys and frequency counts as values.
2. most_common_letter() takes a string as its argument and returns a string of the most common letter(s). In the case of a tie for the most common letter, return them all. This function should call letter_counts().
3. string_count_histogram() takes a string as its argument and returns a list of the unique letters, with each letter being the repeated number of times it appears in the string. This list will then be printed one element per line (as a histogram). This function should call letter_counts().

The following code should be after the functions and inside the block
if __name__ == '__main__':

Assign a sentence of at least 15 characters into a string variable.
Writer 3 print statements with appropriate descriptions that use these functions to create output like the examples.
Sample Program: _marc10is_hw_4_9_5.py

Example Output #1

```
The string being analyzed is: "WAS IT A RAT I SAW"
1. Dictionary of letter counts: {'W': 2, 'A': 4, 'S': 2, 'I': 2, 'T': 2, 'R': 1}
2. Most frequent letter "A" appears 4 times.
3. Histogram:
   AAAA
   II
   R
   SS
   TT
   WW
```

Example Output #2

```
The string being analyzed is: "WWWAS IT A RAT I SAW"
1. Dictionary of letter counts: {'W': 2, 'A': 4, 'S': 2, 'I': 2, 'T': 2, 'R': 1}
2. Most frequent letters ['A', 'W'] appear 4 times.
3. Histogram:
   AAAA
   II
   R
   SS
   TT
   WWWW
```

CS-521 Homework Assignment 4

4.9.6: Create a program that:

- prompts a user for a number
- validates that a number was entered
- re-prompts on error
- converts the number to words using a dictionary
- prints out the converted numbers as words

The program must only have one input command and work for any size positive or negative number.

Decimal point should be converted to 'point'.
If the user enters commas, tell them to try again without the commas.

Example Output #1

```
Enter a number: 123
As Text: One Two Three
```

Example Output #2

```
Enter a number: -123
As Text: Negative One Two Three
```

Example Output #3

```
Enter a number: 1234.76
As Text: One Two Three Four Point Seven Six
```

Example Output #4 – invalid Input

```
Enter a number: 1,000
Please try again without entering commas.
Enter a number: 1 thousand
"1 thousand" is not a valid number. Please try again
Enter a number: 1000.00
As Text: One Zero Zero Zero Point Zero Zero
```

Where to submit?

Click Assignments in the Navigation Area and then click on the title of the assignment to enter the submission area and upload the zip file with your response.