CSC241 – HW4

Reading: Chapter 5

Instructions:

1. **Read the HW guidelines** posted in d2l.depaul.edu > admin.

2. Your solution file MUST be named **hw4.py**. In a comment, include the name(s) of any collaborators.

3. To receive **full credit**, the **names** of files, functions and the **output** must be ***exactly*** as indicated here.

4. **Test your code** by downloading the file **hw4TEST.p**y in the same working folder and typing in the shell:

   ```
   >>> import doctest
   >>> doctest.testfile('hw4TEST.py')
   TestResults(failed=0, attempted=23)
   >>>
   ```

   If you prefer, you can make the test run automatically by adding the following lines at the bottom of your hw2 file/modfule. Every time you run the module, the doctest will be performed.

   ```
   if __name__=='__main__':
       import doctest
       print( doctest.testfile('hw4TEST.py'))
   ```

5. **Copy the** results from the above doctest (it may be longer if you have erorrs) and paste at the top of your submission file. Comment the results out using a multiline comment (triple quotes).

Problems:

1. Write a function `partition` that takes two arguments: a list of strings and a single character. It **prints** the strings in the list that begin with a letter at or after the character (according to dictionary order), one per line. The capitalization of the character and the first letter of the string should not make a difference in whether the string is printed. Empty strings should be skipped. You may assume that the character given as the second parameter is an upper- or lowercase letter. The following shows several sample runs of the function:

```
>>> partition(
['apple','pear','','Orange','Kiwi','','banana'],'g' )
pear
Orange
Kiwi
>>> partition(
['apple','pear','','Orange','Kiwi','','banana'],'G' )
pear
Orange
Kiwi
>>> partition(
['apple','pear','','Orange','Kiwi','','banana'],'w' )
>>> partition(
['apple','pear','','Orange','Kiwi','','banana'],'b' )
pear
Orange
Kiwi
banana
>>>
```

2. Write a function `points` that takes four arguments, The numbers x1, y1, x2, and y2 are coordinates of two points (x1, y1) and (x2, y2) in the plane. Your function should compute:
   a. The slope of the line going through the points, unless the line is vertical
   b. The distance between the two points

   The function should print the computed slope and distance in the following way: if the line is vertical, the value of the slope should be the string 'infinity'. Otherwise the slope and distance will be printed as computed normally. If you don't know how to compute the slope and distance find a reference, for example Wikipedia. The following shows several sample runs of the function:

```
>>> points(0,0,1,1)
The slope is 1.0 and the distance is 1.4142135623730951
>>> points(0,0,0,1)
The slope is infinity and the distance is 1.0
>>> points(3,4,5,6)
The slope is 1.0 and the distance is 2.8284271247461903
>>> points(-12.3,55,99.4,-3.6)
```

```
The slope is -0.524619516562203 and the distance is 126.13821784058946
>>>
```

3. Write a function `numVowels` that takes a string representing a file name as an argument and **returns** the number of vowels (both upper- and lowercase) that appear in the file. For the purpose of this question a vowel is one of a, e, i, o, or u. The following shows how the function would behave for several sample files that are included with the assignment:

```
>>> numVowels('vowels.txt')
21
>>> numVowels('poppins.txt')
418
>>> numVowels('empty.txt')
0
>>>
```

4. Implement a function `numLen` that takes a string `s` and an integer `n` as parameters, and **returns** the number of words in the string `s` that have length `n`. Words in the string are non-space characters separated by at least one space. You may assume that the string has at least one non-space character in it and that the "sentence" does not have punctuation. The following shows several examples of how the function could be used :

```
>>> numLen('This is a test',4)
2
>>> numLen('This is a test',2)
1
>>> numLen('This is a test',3)
0
>>> numLen('The quick red fox jumped over the lazy brown dog',2)
0
>>> numLen('The quick red fox jumped over the lazy brown dog',5)
2
```