22C:16 Homework 6

Due via ICON on Wednesday, March 9nd, 4:59 pm

Submit the solutions to all 5 problems, but we will grade some 3 problems of our choice from your submission.

1. Write a function called `squares` that takes in a parameter called `n` and if `n` is a positive integer, the function returns a list consisting of the squares of the numbers from 1 through `n`. Otherwise (i.e., if `n` is not a positive integer) the function should return an empty list.
   Note: I want this function to work correctly even if the function is called with nonsensical arguments, e.g., `squares("hello")`. Notice that according to the problem statement, the function should return the empty list in such circumstances.

2. A positive integer x is a perfect square if $\sqrt{x}$ is also an integer. For example, 1, 4, 9, 16, 25, 36, 49, 64, 81, and 100 are the perfect squares in the range 1 through 100. Write a function called `perfectSquares1` that takes as parameter a list `L` of integers and returns a list consisting of all the elements of `L` that are perfect squares. For example, if `L` is `[11, 9, 19, 250, 100, -9]`, then the function `perfectSquares1` should return the list `[9, 100]`. The order of the elements in the returned list is not important.
   The algorithm that you *should* use for `perfectSquare1` is the following. First find the maximum integer in `L`. Call this m and generate the list L' of the squares of numbers from 1 through $\lfloor\sqrt{m}\rfloor$. You should use the function `squares` from Problem 1 to do this. Finally, `perfectSquares1` should return all integers in `L` that are also in L'. Make sure that this algorithm makes sense to you before you start your implementation.

3. This problem requires you to solve Problem 2 in a different way.
   (a) Write a one-line boolean function called `isPerfectSquare` that takes a parameter `n` and returns `True` if `n` is a perfect square; the function should return `False` otherwise.
   (b) Write a function called `perfectSquares2` that does exactly the same thing as the function `perfectSquares1`. However, `perfectSquares2` is required to use the function `isPerfectSquare` inside a call to the `filter` function in order to make the desired list.
   Hint: In a correct solution to this problem, `perfectSquares2` will have exactly one line in its body.

4. Write a program that reads some text and outputs the number of times each letter `a` through `z` appears in the text. The user of your program will indicate that they have completed inputting the text by providing an empty line, i.e., typing the `enter` key one extra time after the last line of the text has been input. Your program should treat upper case and lower case letters as being identical and the output that your program produces should be clear and nicely formatted.

5. Write a program that reads some text and outputs the total number of lines, the total number of words, and the total number of characters in the text. The end of the text is specified in exactly the same manner as in the previous problem. To solve this problem you need to know what I mean by a *word*. Here is a definition. With respect to some given text, a word is a contiguous sequence of non-whitespace[1] characters such that:
   (i) the word either starts a line or immediately follows a whitespace character and
   (ii) the word either ends a line or is immediately followed by a whitespace character.

   The character count your program should output should include every character - even whitespaces.

[1] For the purposes of this program, a whitespace character is a blank or a tab.