COP3330 Programming Assignment 4

Super-anagrams and imperfect palindromes

Objective

1. Practice problem solving with C++.

Overview

This is a programming contest style problem. In this assignment, your task is to write a program that determines if word pairs given in an input are super-anagrams and if each word is an imperfect palindrome.

Super-anagrams are words (1) that consist of the same letters, and (2) that the accumulated frequency difference of the letters in the pair of words differs at most 2. For example, “stake” and “takes” uses each letter ‘a’, ‘e’, ‘k’, ‘s’, and ‘t’ once. So they are super-anagrams. “Pumpkin” and “umpire” uses different letters (e.g. with and without ‘k’), so they are not super-anagrams. Words “umpiire” and “umpppire” have the same letters (‘u’, ‘m’, ‘p’, ‘I’, ‘r’, ‘e’), but are not super-anagrams since “umpiire” has two fewer p’s and one more ‘I’: the accumulated frequency difference is 3.

A palindrome is a word that is the same reading forward or backward. Examples include “a”, “aba” and “aabbbaa”. An imperfect palindrome is not a palindrome, but when reading an imperfect palindrome forward and backward, the number of characters that differ is no more then 2. For example, “abc”, “abca”, and “aaabbaab” are imperfect palindromes. “aba”, “abcd”, and “aaabbabb” are not impact palindromes.

The input will contain many lines (from the standard input), each line containing a pair of words to be analyzed. The words only use low-case letters and there is exactly one space between the pair of words. Each word has no more than 20 letters. For each line, your program should output three yes/no answers: the first answers whether the first word is an imperfect palindrome, the second answers whether the second word is an imperfect palindrome, the third answers whether the words are super-anagrams. For example, for the following input:

read dare

stake takes

tofu tofu

aabbba aabbaa

one two

Your program should output:

no no yes

no no yes

no no yes

yes no yes

yes yes no

Submission

The due time for this assignment is June 12 (Wendesday), 2013. 11:59pm.

Tar all of your files for this assignment, which should include two files proj4\_bigin.cpp, and bug\_fixing\_log.txt, name the tar file yourlastname\_firstinitial\_proj4.tar and submit the tar file in blackboard. The bug fixing log file must follow the template given in project 1.

Grading policy

The program must work on linprog. O point for programs with any g++ compiler error on linprog. You will need to track compiling errors that **you fixed by yourselves** in a log for fixed compiler bugs that needs to have at least two entries for 5 points/each entry (10 points max) that are different from those in the bug fixing log in projects 1/2/3.

* Program with no compiler error (20 points)
* Log for fixed compiler bugs (10 points)
* The program gives the correct number and format of the yes/no answers (20 points)
* Correctly check imperfect palindromes (25 points)
* Correctly check super-anagrams (25 points)

Hints

1. Start the project as soon as possible.
2. This is a programming contest type of assignment, so an experienced programmer can do this in 0.5 to 1 hour. But there are groups in our departmental ACM programming contest can spend 4 hours and solve 0 problems – do not assume that you can do this in 0.5 hour.
3. To check super-anagrams, you will need to record the usage frequency for each letter in the words. This can be done with an int array ‘int freq[26];’ You can index into this array from a low-case letter by freq[letter – ‘a’].
4. You will need to create as many test cases as you can manage to thoroughly test your program. The final test cases to be used in the grading will have around 100 cases.