THE UNIVERSITY of NEW MEXICO

Lab 1:
Word Path Using A*

Lab Overview:

Write a Java program taking $2n+1$ command line arguments where n is a positive integer. The first argument is a path to a '/n' delimited dictionary. The remaining arguments are pairs of words. For each input pair of words, the program must find and output a shortest path from the first word to the second word such that:

a) Each word in the path must be a word in the given dictionary.

b) Each word in the path is different from the next word in the path by only a single modification.

c) Allowed modifications are:
   1) Adding one letter to a word (at the beginning, end or anywhere within).
   2) Removing one letter from a word.
   3) Replacing one letter with any other letter used in the dictionary.

Words are not case-sensitive ("greek" is the same as "Greek").

Any character used in a word within the given dictionary is a "letter".

Your program will be tested using the dictionary: `OpenEnglishWordList.txt` on the class website, the given test data on the website and some unknown word pairs.

You may assume the input dictionary is in lexicographical order.

Your program will be tested from the command line with many pairs of words in a single run of the program. Therefore, graph setup time, while important, will have less impact on your overall runtime than will path finding.

Grading Rubric [50 points total]:

[Turn-in: -5 points]: Your program was not turned in correctly (see syllabus).

[Code Style: as much as -10 points]: Your program does not follow the CS-351 hollowed coding standard. More egregious deviants will lose more points (see course website: http://www.cs.unm.edu/~joel/cs351/).

THE UNIVERSITY of NEW MEXICO

[Robustness: as much as -10 points]: For some input, your program gets stuck in an infinite loop or crashes. If your program exits after a word is not found in the dictionary or after processing only some of the input (except as required by the Input Check below).

[Dictionary Path: 2 points]: The first argument is the dictionary path. If a relative path is given, your program must find the path relative to the directory containing your executable JAR file.

[Input Check: 2 points]: If there are less than 3 arguments, or if the first argument is not a file that can be opened, then your program must print an appropriate error message and exit cleanly.

[Output Format: 5 points]: Exactly one line is output for each word pair. The line must either be the space delimited path from the first to the second word or an appropriate error message that includes the word pair.

[Dictionary Check: 3 points]: If one or both search words in a pair is(are) not in the dictionary, then print, on a single line, an error message including the offending word(s).

[Existence Check: 8 points]: If no possible path exists from the first to the second word in an input pair (or the first word is the last argument), then your program must print a single line saying: "NO POSSIBLE PATH:" followed by the word(s).

[Path: 10 points]: When a path exists from the first to the second word in a pair, the program outputs, on a legal path (which may not necessarily always be the shortest).

[Shortest Path: 10 points]: The program always finds and displays a shortest possible path for each given pair of words. If more than one path exists that is no longer than the shortest path, then each such path is equally correct.

[A* Path: 10 points]: The program ***efficiently*** finds the shortest path using the A* algorithm. Total runtime on brown, setebos, sycorax, leda or metis (Dell Precision T3610, Intel Xeon CPU E5-1607 v2 @ 3.00GHz, 16 GB RAM) for correct **`wordPathTestArgs.txt`** output is:

Under 1 minute:	+20 (including 10 Extra Credit)
1 to 3 minutes:	+10 points.
3+ to 10 minutes:	+5 points.
Over 10 minutes:	+0 points.