CS 122 – Homework #4 – "Word Games" – due March 1, 2019

Let's write a couple of programs that would be useful to a Scrabble player. Before you begin, please download the words2.txt file from my Web site. This contains a list of English words in alphabetical order. I will refer to the list of words in that file as the "dictionary." You should store these words in an array or ArrayList. Throughout this exercise, the strings will not be case sensitive. The user should be allowed to enter capital and/or lowercase letters. I recommend that you immediately convert the input to lowercase letters to simplify your comparisons.

The purpose of this program is to use functions to reduce the size of the main() method. Along the way, you will also practice with file input and strings.

Part 1: Finding the max word

This program will be called MaxWord.java. Given a selection of letters, we are interested in the words that can be formed using letters taken from this selection. You do not need to use all the letters of the selection, and in fact this is often impossible (e.g. if the selection contains Q but no U). However, each instance of a letter that appears in the selection may only be used once in the word. For example, if the selection contains only one D, then a solution word cannot contain more than one D. And if the selection contains two Ds, then a solution word cannot contain more than two Ds. Note that a solution word cannot be longer than the selection.

You may assume that the max word will have at least 3 letters. Therefore, when you scan the dictionary, you may skip words shorter than length 3.

The program will take a selection of letters from the command line. In other words we will run the program like this: `java MaxWord <selection>`. For example, `java MaxWord glenrenie` The program should scan the entire dictionary, and print out the longest word that can be formed using this selection. For example, from the selection "glenrenie," the longest word is "engineer." There will often be a tie for the longest word, and in this case, print all the words of this length, in alphabetical order. Print one word per line. Do not print any other text besides these output word(s).

Your implementation should contain a function isMatch() that takes two parameters: a String selection and a String word. The purpose of the function is to see if the word can be formed out of the letters in the selection. For example, "costume" is obtainable from the selection "toscuinme", but "cop" is not. This function should return boolean. The function should be called from within a loop that traverses the entire dictionary of possible words.

This program will be called Substring.java. In a Scrabble game, it is usually necessary to form a word using existing letters on the board. You will write a program to help a user find all possible words that contain a given substring. However, the substring does not need to be completely described. For example, we might want to find all words in the dictionary containing the substring "dog". Or maybe we just want all the words containing a 'd' followed by a 'g' with exactly one intervening space for a letter. This substring could be encoded by using a symbol such as the '?' as a wild card representing one mandatory, but unspecified, character, like this: "d?g". For example, given the substring "u?v?yi," one solution is "surveying."

Write a program that takes a substring from the command line, possibly including wild card characters '?'. The program will be run as: java Substring <pattern>. For example: `java Substring d?g` Print out all words in the dictionary that contain this substring. Print one word per line, in alphabetical order. Do not print any other text besides these output word(s).

Your program should contain a function called isMatch(), similar to the function you wrote for the first program. It should take two parameters, a String substring, and another String word. The purpose of the function is to see if the word contains the substring. The substring may contain wild cards '?'. Each '?' represents one mandatory letter, but it can be any letter. For example, if the word is "costume" and the substring is "o?t?m", then the function should return true. This function should be called from within a loop that traverses the entire dictionary.