CSE116 Problem Set #1

For each of the following problems you should start by writing several JUnit tests that will verify correct behavior. Once you have written at least four distinct tests for each of the problems, begin to implement solutions to each of the problems. Be sure to run your unit tests often. For all these methods, when it comes time to implement them, the only methods you may call on a String are charAt(int) and length().

1. Define a method named count_e that accepts a String as input and returns the number of times the char 'e' occurs, as an int.

2. Define a method named count_one_char that accepts a String and a char as input and returns the number of times the char occurs in the String, as an int.

3. Define a method named count_two_chars that accepts a String and two chars as input (i.e. it has three parameters in total). Assuming that the two char parameters are named c and d, define this method so that it returns, as an int, the number of times the char c occurs in the input added to the number of times the char d occurs in the input.

4. Define a method named count_chars_in_String that accepts two Strings as input. Assuming that the two parameters are named s and characters, define this method so that it returns, as an int, the number of times any of the chars in the second String (characters) occur in the first String (s).

5. Define a method named isPalindrome that accepts a String as input and returns, as a boolean, whether the input is a palindrome or not. A palindrome reads the same way forward and backwards. Remember that uppercase and lowercase characters are distinct.

6. Define a method named hasAdjacentRepeats that accepts a String as input and returns, as a boolean, whether the input has two adjacent characters which are identical. For example "book" has 'o' as an adjacent repeat, whereas "oboe" does not.

7. Define a method named firstIndexOfAdjacntRepeats that accepts a String as input and returns, as an int, the lowest index at which two (or more) adjacent characters can be found in the input, if there are adjacent repeats. Otherwise the method must return -1. For the input "book" the answer must be 1. For the input "oboe" the answer must be -1.

8. Define a method named replaceVowels that accepts a String as input and returns a new String which is just like the input String except that all vowels have been replaced by '*'. For example, for input "book" the returned value must be "b**k". Assume that the characters 'a', 'e', 'i', 'o' and 'u' (and their uppercase equivalents) are vowels.

9. Define a method named replaceCharacters that accepts two Strings (call them input and targets) and a char (call it c) and returns a new String identical to input except that any character from targets is replaced by c. For example, replaceCharacters("banana","bn",'?') must return "?a?a?a".