CSCI-15 Lab 8, due 12/6/18

Please work in groups of two on this.

A palindrome is a string of characters that reads the same way going forward and backward, ignoring white space and punctuation. Write a program that reads a series of lines of text and tests each line, using both an iterative and recursive IsPalindrome() function, to see if it is a palindrome or not. In your test, ignore upper- or lower-case differences, whitespace characters (space or tab) and punctuation.

Write a main function that loops, getting lines of text as strings from input, until it reads the string "quit". You may use direct file I/O or redirect input from a file if you want. For each string `<string>` read, and each call to the two functions, if `<string>` is a palindrome, print

```
|<string>| is a palindrome
```

and if not, print

```
|<string>| is not a palindrome.
```

Use either vertical bars "`|`" or angle brackets "<>" to delimit the string when you echo-print it.

You may use either C-strings or C++ string objects for this assignment. The character type library (`#include <cctype>`) has a function `isalpha()` that takes a character (actually it takes an int, but it treats it like a character) and returns true if the argument is alphabetic and false otherwise. It also has a function `tolower()` that takes a character and, if it is upper case, shifts it to lower case and returns it, otherwise returns it unchanged.

This program should be less than one page of code.

For example, a test run taking input from `cin` might look like this:

```
Enter strings, "quit" to quit
adam
|adam| is not a palindrome
|adam| is not a palindrome
madam
|madam| is a palindrome
|madam| is a palindrome
Madam, I'm Adam.
|Madam, I'm Adam.| is a palindrome
|Madam, I'm Adam.| is a palindrome
A man, a plan, a canal, Panama!
```

```
|A man, a plan, a canal, Panama!| is a palindrome
|A man, a plan, a canal, Panama!| is a palindrome
not a palindrome
|not a palindrome| is not a palindrome
|not a palindrome| is not a palindrome
quit
```

Hints: The client will just pass the string by reference (a C-string, since it is an array, is automatically passed by reference). The recursive version of the function the client calls should use a recursive helper that takes the string (by reference if needed) and the low and high indices of the part to examine. (Why shouldn't the client call the helper directly?)

For the recursive version, if the low and high indices are the same, or low is higher than high, then the string is a palindrome. If the character at the low or high indices isn't a letter, skip it and try again recursively with the next lower or higher index. If both characters are alphabetic and different (ignoring case) then the string is not a palindrome. If the characters at the low and high indices are alphabetic and the same (ignoring case) you don't know the result yet, so try recursively with low+1 and high-1 indices.

Do not alter the string in the functions. Do not pass the string by value.

The iterative version may not recurse and the recursive version may not loop.