Programming Assignment #7

Due: Fri, Dec 9

IMPORTANT NOTE: No late submissions will be accepted for this one. Dec 9 is the ABSOLUTE DEADLINE, in order to leave appropriate time for grading

This assignment contains two exercises based on linked list and stack classes

---

Before You Start

Here are the List and Stack examples that we looked at in lecture class (From Deitel edition 5). Here are links to copies of the files:

- listnode.h
- list.h
- stack.h

You will need these files for both exercises.

Exercise 1

Filename: **`palindrome.cpp`**

Write a program that uses a stack object to determine if a string is a palindrome (i.e. the string is spelled identically backward and forward). The program should ignore spaces and punctuation.

Go ahead and start your program by reading in a C-style string from standard input, using the getline function. You may assume a limit of 100 characters on the string (although this can be written, with a little more effort, to accept any size string). Your algorithm must make use of a stack (of type char). Use the Deitel implementation of the Stack from "stack.h" (you **don't** need to change this file).

Ignore spacing and punctuation, as well as special characters and digits (i.e. only count letters as part of a palindrome, and account for upper/lower case. For example, 'B' and 'b' are matching letters).

Sample runs: (user input underlined)

```
  Please enter a string:
  > ABCDEFGHGFEDCBA

  "ABCDEFGHGFEDCBA"  IS a palindrome


  Please enter a string:
  > The quick brown fox

  "The quick brown fox"  is NOT palindrome
```

```
Please enter a string:
> Cigar? Toss it in a can. It is so tragic.

"Cigar? Toss it in a can. It is so tragic."  IS a palindrome
```

Want some palindromes to test with? Try this site

General

Since you are reading data into a C-style string to begin, you may use any of the libraries `<iostream>`, `<cstring>`, and `<cctype>`, if you like. (The first, of course, for I/O, and the other two, since they deal with C-style strings and characters).

Exercise 2

Modify the List class (file `list.h` so that it has two more functions, which will allow inserts and removes from anywhere in the linked list. Your functions should be called:

- `insertMiddle`
- `removeMiddle`

Your functions should have all the same features as the given `insert` and `remove` functions, except that yours each have one extra parameter. The second parameter on each of your functions should be of type `int`, representing the **position** at which to insert (or delete). Sample calls for a list of integers:

```
L.insertMiddle(345, 5);       // attempts to insert the value 345
                              //  as the 5th item in the list

L.removeMiddle(x, 10);        // attempts to delete the 10th item in the
                              //  list and captures its value into x.
```

For `insertMiddle`, if the position number is larger than the number of items in the list, just insert the item at the **back**. If it's too small (i.e. 0 or less), insert at the **front**. For `removeMiddle`, return false if the position is invalid (without removing anything).

I've modified the menu program of Figure 21.5 so that it adds in two more menu options for testing these features. You can use it to test your class: menu7.cpp

---

Submitting:

Submit the files:

```
palindrome.cpp
list.h
```