lab08 : Anagrams, palindromes: Strings and recursion

num	ready?	description	assigned	due
lab08	true	Anagrams, palindromes: Strings and recursion	Tue 11/26 09:00AM	Thu 12/05 11:59PM

Goals of this lab

This lab will have you do programming exercises with C strings, string class objects, recursive functions, and Makefiles. You will also get more practice writing programs from scratch with little skeleton code.

Step 1: Getting Started

1. Decide if you are working alone, or working in a pair. Pair programming is OPTIONAL for this lab.
2. If you are working as a pair, remember to add your pair partner on gradescope.
3. Go to github and create a git repo for this lab following the naming convention specified in previous labs (this step carries style points, see our feedback on previous labs to understand what we are looking for). If you are working with a partner only one of you needs to create the repo.
4. If you are working with a partner and you are the one who created the github repo, add your partner as a collaborator on the repo
5. Driver, log on to your CSIL account.
6. Open a terminal window and log into the correct machine.
7. Change into your CS 16 directory
8. Clone your github repo in the ~/cs16/ directory. Then cd into your repo directory.

Note: Remember to push your work to github at the end of EVERY work session. That way, both partners always have access to the latest version of the code even if the code is being developed on one partner's CoE account.

Step 2: Getting the starter code and writing the programs

Clone your github repo in the ~/cs16/ directory. Then cd into your repo directory.

```
cd ../lab08_gaucho_ally
```

Copy the code starter code for lab08 using the following command.

```
cp /cs/faculty/dimirza/cs16/labs/lab08/* ./
```

You should see the following files:

```
[dimirza@csil-01 lab08]$ls
linkedListFuncs.cpp  README.md              strFuncs.cpp  tddFuncs.h
linkedListFuncs.h    recLinkedListFuncs.cpp  strFuncs.h
linkedList.h         recLinkedListFuncs.h    tddFuncs.cpp
```

This lab will have you write two functions that are specified in strFuncs.h and two functions that are specified in recLinkedListFuncs.h. You must implement these functions in strFuncs.cpp and recLinkedListFuncs.cpp. You must follow the instructions carefully. It is not enough to pass the gradescope check as the instructor and the TAs *will* be checking your submitted program files for style.

Program to find if two strings are anagrams

In the file strFuncs.cpp, write a function called isAnagram that takes two strings as arguments and returns a boolean true if the two strings are anagrams, otherwise it returns false. The function should not be case sensitive and should disregard any punctuation or spaces. Two strings are anagrams if the letters can be rearranged to form each other. For example, "Eleven plus two" is an anagram of "Twelve plus one". Each string contains one "v", three "e's", two "l's", etc. Similarly "Rats and Mice" and "in cat's dream" are anagrams of each other. You may use any of the C string library or string class functions to complete this code. You may **not** use built-in C++ functions that we have NOT discussed in lecture. You must follow a TDD style of coding. Write your own test code in a separate file and write a Makefile to compile the code.

Program to check if an input string is a palindrome

In strFuncs.cpp you are asked to a function to check if a string is a palindrome. For example: "redivide" is not a palindrome, while "detartrated" is a palindrome. Read the instructions in the file carefully to understand the constraints specified for the function. Ignore case when comparing characters of the string.

```
bool isPalindrome(const string s1);
```

The above function takes a C++ string as input and returns true if an input string is a palindrome and false if it is not. You can do this by checking if the first character equals the last character, and so on. You may choose a recursive implementation in this case, although it is not required. If you chose a recursive implementation you may or may not choose to write a helper function. You won't need a helper function is you used the substr (substring) function appropriately in your recursive calls.

Program to recursively find the sum of elements of a linked list

```
int recursiveSum(Node* head);
```

In recLinkedListFuncs.cpp you are asked to reimplement the sum function from last week, this time recursively. If there are no elements of the list, return the value 0;

Program to recursively find the largest value of a linked list

```
int recursiveLargestValue(Node* head);
```

In recLinkedListFuncs.cpp you are asked to reimplement the largestValue function from last week, this time recursively. For this function, you may assume that the linked list contains at least one value.

Step 3: Submit

Push all your code to github. Then submit your code on gradescope.

Make sure you add your partner as a collaborator.

Step 4: Check your submission results

You may submit this lab multiple times. You should submit only after local compilation does not produce any errors and runs as expected. The score of the last submission uploaded before the deadline will be used as your assignment grade.

Step 5: Done!

Remember that we will check your code for appropriate comments, formatting, and the use of required code, as stated earlier.

If you are in the Phelps lab or in CSIL, make sure to log out of the machine before you leave. Also, make sure to close all open programs before you log out. Some programs will not work next time if they are not closed. Remember to save all your open files before you close your text editor.

If you are logged in remotely, you can log out using the exit command:

```
$ exit
```