Practical 1: Fun with Strings

Due date: 11:59pm, 15 March 2019

1 General Instructions

All submissions for the practical assignments should be under version control. If you are not familiar with SVN, please check the entry "How to Use SVN" on MyUni.

1.1 SVN

For all your assignments, first of all, you need to create a directory under version control:

```
svn mkdir --parents -m "comments" https://version-control.adelaide.edu.au/svn/aXXXXXXX/[year, e.g. 2019]/
[semester, e.g. s1]/adds/assignmentK/
```

In the above command, aXXXXXXX should be your student ID. The directory path needs to be exactly "YYYY/sX/adds/assignmentK", where "YYYY" is the year, "sX" can be s1 or s2 for semester, and "K" is the assignment number. for the first assignment, and the first semester of 2019, this would be:

```
svn mkdir --parents -m "comments" https://version-control.adelaide.edu.au/svn/aXXXXXXX/2019/s1/adds/assignment1/
```

To check out a working copy, type:

```
svn checkout https://version-control.adelaide.edu.au/svn/aXXXXXXX/YYYY/sX/adds/assignmentK/ local_folder_
name_for_assignment_K/
cd local_folder_name_for_assignment_K
```

Suppose your submission contains three files: main.cpp, myclass.cpp, and design.pdf. After creating these files, you may add them to version control as follows:

```
svn add main.cpp
svn add myclass.cpp
svn add design.pdf
```

Commit the files to SVN:

```
svn commit -m "Adding ADDS prac 1 files"
```

SVN helps keeping track of file changes (over different commits). You should commit your work early and often.

1.2 Websubmission

The submission of your practical assignment should be made via the web interface `https://cs.adelaide.edu.au/services/websubmission/`. The submission steps should be self-explanatory. Simply choose the correct semester, course, and assignment. The websubmission system will automatically fetch the latest version of your work from your SVN repository (you may also choose to submit an older version). Once your work is submitted, the system will launch a testing script. Click "View Feedback" to view the testing results (don't worry about the 0 mark). If your code fails some test cases, then most likely your code contains bugs. You are welcome to resubmit for as many times as you wish. (Your last submission determines whether it is considered late or not.)

The test script will compile your code using `g++ -o main.out -O2 -Wall *.cpp`. It is your responsibility to ensure that your code compiles **on the university system**.[1]

2 Problem Description

The first practical is intentionally made easy. The main objective is for you to get started on using SVN and the websubmission system. This practical will also test your knowledge of C-Strings, dynamic arrays, and pointers.

This assignment deals with strings. You are encouraged to try different implementation methods (e.g., try both std::string and C-string).

A phrase is called a *Palindrome* if it is the same once reversed. Reversing "Race fast, safe car." gives us ".rac efas ,tsaf ecaR", which is actually the same as the original phrase *if we ignore non-alphabetical characters and cases*. Phrases like these are called Palindromes.

You are asked to create a Palindrome class that stores a phrase and supports three methods:

- removeNonLetters: remove all non-alphabetical characters from the phrase
- lowerCase: change all letters in the phrase into lower cases
- isPalindrome: return whether the phrase is a Palindrome

You are asked to also create a main function. The main function takes as input a phrase (one line of characters, containing letters, spaces, digits, and punctuations). Hint: try `getline` function. Your main function should output whether the phrase is a Palindrome.

Sample input 1: Race fast, safe car.

Sample output 1: Yes

Sample input 2: Bo1b

Sample output 2: Yes

Sample input 3: ADDS is fun

[1] g++ has too many versions, so being able to compile on your laptop does not guarantee that it compiles on the university system. You are encouraged to debug your code on a lab computer (or use SSH).

Sample output 3: No

Your submission should contain at least `main.cpp`, `Palindrome.cpp`, `Palindrome.h` and `design.pdf`. Your submission will be tested using the above sample test cases (as well as a few additional hidden test cases).

3 Marking Scheme

Each practical assignment is worth 3% of your final mark, except for the last one (prac 8) which worth 4%.

You may submit your work before your practical session in Week 2 to get marked by your tutors so that you can improve your answer based on the feedback and resubmit before the due date.

Every practical assignment is marked out of 6.

- Design (2 marks):
  - Diagram of central classes and explanation of core functions (1 mark)
  - Details of your own test cases/schemes (1 mark)
- Style (2 marks):
  - Proper usage of C++ language elements and paradigm (1 mark)
  - Comments on non-trivial code blocks and functions (1 mark)
- Functionality (2 marks):
  - Passing public test cases (1 mark)
  - Passing hidden test cases (1 mark)

Your final mark for this assignment will be based on your latest submission in the web-submission system. If you have already got a higher mark, it will not be replaced with a lower mark.