CSCE 221 Assignment 3 Cover Page

First Name Zhiyang Last Name Zeng UIN 720005338

User Name zhiyangzeng E-mail address zhiyangzeng@tamu.edu

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero, read more on Aggie Honor System Office website: `http://aggiehonor.tamu.edu/`

Type of sources				
People				
Web pages (provide URL)	Stackoverflow.com			
Printed material	Lecture slides, textbook			
Other Sources				

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.
On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work .

Your Name

Date

10/13/2015

Program Description
Traditional array structure can be costly when its size has to be constantly changing. A doubly linked list can be used to add and delete new data at a constant rate.

Purpose of the Assignment
The program is to implement doubly linked list as a template and use it to store phonebook data with names, UIN, and phone number into the list. Users can search the database using last name, first name, and UIN.

Data Structures Description
The new data structure used in this assignment I learned is doubly linked list. The phonebook is a custom structure that stores a student's last name, first name, UIN and phone number. Student with the same last name initial are stored in the same doubly linked list. Records within the doubly linked list is ordered by last name, first name, and UIN. All 26 linked list, one for each alphabet, are stored in a vector of size 26.

Algorithm Description
Because doubly linked list cannot access element by index, the only algorithm occurred in this program is linked list traversal, which is linear search with O(n). The insertOrderly function has a runtime function $F(n)=(n-2)*1+6= n+4$, which is O(n).

Program Organization and Description of Classes
There are 4 parts to the program, each stored in a separate folder.
The SimpleDoublyLinkedList part contains one .cpp file with struct declaration, implementation, and a main tester program.
The DoublyLinkedList part contains one header with DListNode class declaration and DoublyLinkedList class (used to store nodes into a list) declaration. It also contains a .cpp file to implement the classes, and a separate main.cpp tester program.
The TemplateDoublyLinkedList part contains one header file with class declaration and implementation that are the same in DoublyLinkedList, this time with template added to work with generic types. TemplateMain.cpp is the tester program.
The Phonebook part contains the TemplateDoublyLinkedList header file and a Record.h header file containing the structure to store individual student records. PhoneBook.txt is the file containing student record to be read and stored into the program. Main.cpp is the tester program.

Instructions to Compile and Run
SimpleDoublyLinkedList: "make all" to compile, "./simpledoublylinkedlist" to run.
DoublyLinkedList: "make all" to compile, "./main" to run.
TemplateDoublyLinkedList: "make all" to compile, "./templatemain" to run.
Phonebook: "make all" to compile, "./main" to run.

Input and Output Specifications
The program reads file from PhoneBook.txt. The program will throw exception "Empty Doubly Linked List" when trying to call first(), last(), or remove functions on an empty list. It will cout "File not found" if the txt file name could not be found or loaded. It will throw exception

“Invalid choices” for invalid input choices to interact with data. Other exceptions are handled with “Something is wrong”.

Logical Exceptions
All logical exception of all users cases I can think of are handled by exceptions. The output of exceptions are discussed in the previous section.

C++ object oriented / generic programming features
Template is used in the phonebook program and template doubly linked list. Not many other object oriented feature is used, except keeping the class in a separate file from the implementation.

Tests
The provided tester programs have done many different testing on the doubly linked list such as copy constructor, “=” overloading, insert, delete. Additionally, insertOrderly has been tested to make sure that data of any value can insert at the right place. The phonebook program’s search function has been tested multiple times on many different names to make sure that student with the exact same name but different ID can be differentiated.