**CSCE 221 Assignment 3 Cover Page**

First Name: **Kaivan** Last Name: **Shah** UIN: **522006920**

User Name: **kks6920**  E-mail address: **kaivan29@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 | Kyle Almeida | Brandon |  |
| Web Pages (provide URL) | http://www.cplusplus.com/ |  |  |
| Printed Material | Data Structures and Algorithms in C++ |  |  |
| Other Sources | PowerPoint |  |  |

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 Kaivan Shah Date 10/10/14

**Program Description**

Part 1:

(i) I had to define the functions insert\_after , insert\_before, delete\_after, and delete\_before for a simple doubly linked list class. The simply doubly linked list class helps us understand how the nodes work and how to insert them into a list.   
(ii)For Double Linked List class I had to define the copy constructor, overload the assignment operator(=) and the output operator (<<). This helped me iterate through the Doubly Linked List class using header and trailer.   
(iii) In TemplateDoublyLinkedList class I had to create a template for more generic programming. Later I realized that we are going to create DoublyLinkedList of Records.

Part 2:

I used the above defined classes to implement a phone book of students which stores the last name, first name, 9-digit UIN, and phone number of each student. The phonebook had to be read in from a text file and sorted orderly alphabetically. Also, a search function is created for the user search for a student’s records.

**Purpose of the Assignment**

The purpose of this programming assignment is to learn about data structures and algorithms. Implementing these functions provided a deeper understanding of templates and the functionality of double linked lists. I learned how double linked lists works with its nodes through the help of the header and the trailer.

**Data Structures Description**

The main data structure in this assignment is the double linked list.

For Part 1:A double linked list contains a header which helps us identify the front and a trailer for the back of the list. Every element in the list contains an object and two pointers, previous and next, that point to the previous element and next element respectively. To add or remove the objects of the list we could do it using the pointers as the elements in the list could not be accessed using an index.

For Part 2: I created a class called Records which stored a student’s last name, first name, UIN, and phone number. Then I created a vector of 26 Lists having the nodes as Record that saved the Names Alphabetically.

**Algorithm Description**

where n is the size of the list.

*1. insert\_before(d)* – inserts d before this node - Big O(1)  
A new node is created with a "previous" pointer to the previous node, and a "next pointer to the current node. the previous node's "next" pointer is assigned to the created node and the current node's "previous" pointer is assigned to the created node.

*2.* insert\_after(d) – inserts d after this node - Big O(1)  
A new node is created with a "previous" pointer to the current node, and a "next pointer to the next node. the current node's "next" pointer is assigned to the created node and the next node's "previous" pointer is assigned to the created node.

3. delete\_before() – deletes the node before this node - Big O(1)  
The function first checks if a previous node exists. The second previous node's "next" pointer is assigned to the current node, the previous node is deleted, and the current node's "previous" pointer is assigned to the previous node's "previous" pointer

4.delete\_after()– deletes the node after this node - Big O(1)  
The function first checks if a next node exists. The second next node's "previous" pointer is assigned to the current node, the next node is deleted, and the current node's "next" pointer is assigned to the next node's "next" pointer.

5. copy constructor – copies the doubly linked list – Big O(n)  
Creates a pointer to iterate through the argument that use "insertlast" to add them to the newly constructed DLL.

6. assignment operator – assigns the same values as the doubly linked list – Big O(n)  
Creates a pointer to iterate through the argument that use "insertlast" to add them to the newly constructed DLL.

7. output operator – prints the list to the screen – Big O(n)  
Checks to see if the DLL is empty, if it isn't it creates a pointer to iterate through the DLL, outputting each element as it does so.

8.less-than operator – compares two Records and is true if the Record on the left is smaller alphabetically – Big O(1)

9.insertOrderly – inserts an object to the correct position if the linked list is sorted– Big O(n)

**Program Organization and Description of Classes**

I used a Template Doubly Linked List class and it is a friend class of the DListNode class.

I created a Record class which stored student info. In the main function, I used a Doubly Linked List of Records.

**Instructions to Compile and Run your Program**

Compile the Simple Doubly Linked List with **make** and run it with **./SimpleDoublyLinkedList**

Compile the Doubly Linked List with **make** and run it with **./Main**

Compile the Template Doubly Linked List with **make** and run it with **./TemplateMain**

Compile the phonebook with **g++ -std=c++11 -o Main \*.cpp** and run it with **./Main**

To run them use: ./a.out

**Input and Output Specifications** Text files with the format as the give text file are required. Record tabs the first name and uses each element on a new line. When the last name is inputed and it matches, all the details are displayed. If not then first name and if not then the uin are inputed into the program, and simply the details are displayed.  
 **Logical Exceptions**

When I display the phonebook I was not able to get the blanked lists to disappear which instead printed blank lines.

**C++ object oriented or generic programming features**

The Doubly Linked List Class is a friend class to the DListNode class which means Doubly Linked List can access the private and protected members of DListNode. I made a Generic Doubly Linked List using Template so that it can store any desired type into the list.

**Tests**

**How the Program runs:** Enter the name of the textfile you want to read.

PhoneBook.txt

Raw File:

The phoneBook has the following contents:

Last name - First Name: Andrews-Edna

UIN:528320876

ph\_no:4352514822

Last name - First Name: Arenas-Edward

UIN:239924731

ph\_no:2525976612

Last name - First Name: Autry-Richard

UIN:527646269

ph\_no:6028236739

Last name - First Name: Latham-Mary

UIN:174485583

ph\_no:2156902061

Last name - First Name: Lawrence-Adelle

UIN:606880340

ph\_no:4154068779

Last name - First Name: Leblanc-Fred

UIN:253174074

ph\_no:6782391146

Last name - First Name: Lee-Rolando

UIN:680385098

ph\_no:7752476489

Last name - First Name: Lester-Mozell

UIN:173503264

ph\_no:5706145308

Last name - First Name: Lile-Francis

UIN:237818062

ph\_no:7044943520

Last name - First Name: Lopez-Felecia

UIN:524085043

ph\_no:3036105461

Last name - First Name: Weaver-Andrew

UIN:460497442

ph\_no:2817601558

Last name - First Name: Whitney-Floyd

UIN:367196315

ph\_no:2313697993

Last name - First Name: Whitney-Floyd

UIN:367196315

ph\_no:2313697993

Last name - First Name: Wilham-Bernice

UIN:008641309

ph\_no:8027701514

Last name - First Name: Wiseman-Kristi

UIN:311708896

ph\_no:8122398910

Last name - First Name: Wiseman-Mary

UIN:150521209

ph\_no:7324463703

Last name - First Name: Wiseman-Mary

UIN:224376947

ph\_no:7037213439

Last name - First Name: Wiseman-Mary

UIN:464996747

ph\_no:3253740973

Last name - First Name: Wiseman-Paul

UIN:347075568

ph\_no:2179340158

Let us search through the phoneBook

Enter the last name:

WiseMan

Please Enter the First name:

mAry

Please Enter the UIN:

224376947

The details of the person that you asked for:

Last name - First Name: Wiseman-Mary

UIN:224376947

ph\_no:7037213439