CSCE 221 Cover Page
Programming Assignment #1
Due February 5 by midnight to CSNet

First Name Shaeeta Last Name Sharar UIN 822006676

User Name ssharar E-mail address ssharar@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. According to the University Regulations, Section 42, scholastic dishonesty are including: acquiring answers from any unauthorized source, working with another person when not specifically permitted, observing the work of other students during any exam, providing answers when not specifically authorized to do so, informing any person of the contents of an exam prior to the exam, and failing to credit sources used. Disciplinary actions range from grade penalties to expulsion read more: Aggie Honor System Office

Type of sources			
People			
Web pages (provide URL)			
Printed material	Textbook: Data Structure and Algorithims in C++		
Other Sources	Slides: Analyzing Algorithims	Piazza	

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 Shaeeta Sharar Date 03/12/2015

Report
Shaeeta Sharar

Part 1

Program Description

The problem of Part one was to implement three different versions of doubly linked lists. The first was a simple version made out of a struct of nodes. The second was a more complicated class of doubly linked list which included a class for node as well. The final was a template implementation of doubly linked lists. The purpose of the assignment was to familiarize ourselves with the data structure of linked lists and to better understand using pointers (such as in nodes).

Data Structures Description

The data structure that was focused on was the doubly linked list. We implemented this structure by creating a class for linked lists and its elements: nodes. We then used pointer manipulation to access nodes and the elements held in them. Generic programming was also implemented, thus allowing for a more varied use of the class created.

Algorithm Description

The first part, the simple doubly linked list called for the implementation of insert and delete functions. Both insert functions (insert before and insert after) were implemented by first creating a new node with a value of the one provided then setting the pointers of the nodes before and after it to the new node. The delete functions of delete before and after were implemented in a similar fashion. The nodes around the one being deleted were set so that their pointer pointed to each other. Then the original node was delted using a destructor. All of these function were of constant O(1) complexity.

In the more complex doubly linked list classes, the copy, assingment and output operators were implemented. The copy constructor was implemented through the use of a while loop that ran through the linked list being copied and set its nodes to the one being called. The assignment operator was done in a simplar way, firstly a check was made to prevent self assignment. Then the old list was deleted and all elements were put into the new list. The output operator also used a while loop to run through each node and output its value. These were of complexity O(n) due to their loops.

Program Organization and Description of Classes

The simpe linked list contained a struct of nodes. This class contained the insert and delete functions used in a linked list. This implementation was simply a collection of nodes pointing to each other, it was not a full linked list. The last two implementations were of a more advanced doubly linked list that included two classes: the node and the linked list. The linked list was the main class. It contained the usual member functions such as accessors, insertion and deletion functions. The node class contained the element and pointers to the other nodes. The linked list class was a sort of shell for the node class.

Instructions to Compile and Run your Program

To compile the Simple Doubly Linked List, go into the directory labled SimpleDoublyLinkedList. Once in the directory, simply use the "make" and "main" commands to run as specified in the makefile. Likewise to compile the Doubly Linked List, go into the directory labled DoublyLinkedList. Once in the directory, simply use the "make" and "main" commands to run as specified in the makefile. This same process is used for Template Doubly Linked List though it should be noted that the directory must be the TemplateDoublyLinkedList directory.

Input and Output Specifications

For the simple doubly linked list, there are no input specifications. The output should be displayed to the command line. Likewise, neither the template doubly linked list nor nongeneric doubly linked list require any inputs and the outputs are displayed to the screen.

Logical Exceptions

In the algorithims written, there are checks to ensure that that there are no null pointers being accessed and causing errors. Otherwise failure to initialize a header and trailer in the simple doubly linked list could cause errors. This as well as using non integer types. In the doubly linked list, not using integer types also causes errors. This is fixed in the template version, which is used for generic programming.

C++ object oriented or generic programming features

The template version of doubly linked list is implemented through the use of generic programming features. A template type "T" is used in place of other

types. Each function including constructors is reworked to work with the templates. The use of multiple classes in the doubly linked list and template version is also an example of inheritance. The class doubly linked list uses the class nodes as a storage structure for its data.

Tests

These are implemented in the main file of each type.

Part 2

Program Description

The purpose of this part was to implement a user defined type of "SparseMatrix" which used features of vectors and linked lists to store its data. The program was created using the standard library vector and the template doubly linked list created in the last part.

Data Structures Description

• Write which data structures have you learned about by doing in this programming assignment.

• Write about implementation of data structures using the C++ programming language.

The data structures used in our sparse matrix were doubly linked lists and vectors. These together creatred our SparseMatrix data structure. Each index of the vector stored doubly linked lists. This was used at a way to represent a 2d matrix, paritcularly a sparse one. Classes were used to implement this function as well as a lot of pointer manipulation and operator overriding.

Algorithm Description

• Write about algorithms that you used to solve the problem and describe their implementations.

• Analyze algorithms according to assignment requirements.

Set:
Get:
() Operator:
Operator +:

Operator -:

Program Organization and Description of Classes

• Put class definitions in a header file .h and the

ir implementation in a .cpp file

• List all the classes or interfaces you used and show relation between them.

– Main Class

– Classes you used to implement the main data structures

– Exceptions classes

– Other classes

Instructions to Compile and Run your Program

• How to compile: (specify the directory and names of files)

• How to run: (Specify the name of a file to run)

Input and Output Specifications

• Format of data from keyboard (if applicable). In case the program requires a sequence of items, specify if you have to introduce one item per line or all items on the same line. Make an example. • Format of data from file (if applicable) • Cases in which your program crashes because of wrong input (eg, wrong file name, letter instead of number, or the program expects 10 items and it only finds 9.) • Cases of wrong input that you catch with Exceptions.

Logical Exceptions

List cases in which your program crashes or when you catch on exception due to logical execution problems. For example division by 0, or deletion of an item from an empty list. • Cases in which the program crashes • Cases in which you catch exceptions

C++ object oriented or generic programming features

List and describe the C++ object oriented or generic programming features you used in this assignment: • Inheritance • Polymorphism • Templates

Tests Test the logical correctness of your program. Do not test the menu. Describe the cases, and copy and paste the execution of your prog