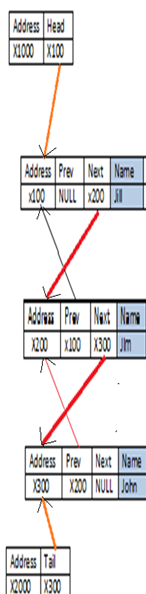
Write a program that reads a list of students (first names only) from a file. It is possible for the names to be in unsorted order in the file but they have to be placed in sorted order within the linked list.

The program should use a **doubly linked list**.

Each node in the doubly linked list should have the student’s name, a pointer to the next student, and a pointer to the previous student. Here is a sample visual. The head points to the beginning of the list. The tail points to the end of the list.



When inserting consider all the following conditions:

if(!head){ //no other nodes

}else if (strcmp(data, head->name)<0){ //smaller than head

}else if (strcmp(data, tail->name)>0){ //larger than tail

}else{ //somewhere in the middle

}

When deleting a student consider all the following conditions:

student may be at the head, the tail or in the middle

Below, you will find a sample of what the file looks like. Notice the names are in unsorted order but the information placed in the linked list (above visual) is in sorted order. The name of the file should be “input.txt”.

In the text file, the word delete followed by a name, should delete the node with that specific student’s name from the doubly linked list. If the name is not found, then nothing is deleted.

(NOTE: The above visual represents only the first three lines from the text file below.)

Jim

jill

John

**delete jill**

**Bob**

**Jack**

**delete jim**

At the end of the program, traverse through the contents of the linked list in both ascending and descending order, using the doubly linked list, and write the contents into the file output.txt. For example, given the above list, here is the sample display:

Bob

Jack

John

=============

John

Jack

Bob

Doubly Linked list (names)

1. Do you have any suggestions about how I should start this lab?

A: Make sure that you draw out the nodes on a piece of paper so that you know exactly what is being connected to what. If you try to do this in your head, then more than likely, you will not do well.

1. Do I need to use some sort of sort function to sort the nodes?

A: No, do not use any of the usual sorting algorithms. It will complicate life. Once you create a node, you need to decide where it goes in the doubly linked list. It can go in the beginning which affects the head. It can go at the end which affects the tail. It can go in the middle which means it doesn’t affect either and finally, if it is a duplicate, you can fully ignore it.

Once you create your node, start traversing through your linked list. A variable like curr will point to the node in your linked list that you are on. Compare the name in curr with the name in the newly created node. If the name in curr node is bigger, then you have found the spot to insert your node. If the name in curr node is smaller, then you continue to move to the next node in your list. If you have reached the end of the list and the name in curr node is smaller, then you have to insert at the end and update the tail. If the name in the curr node is bigger and you are at the beginning of your list, then you have a new head. If the names are the same, then you can just get out because you have a duplicate

if(!head){ //no other nodes

}else if (strcmp(data, head->name)<0){ //smaller than head

}else if (strcmp(data, tail->name)>0){ //larger than tail

}else{ //somewhere in the middle

Use a loop to traverse through the list to find the location of where you need to insert

}

1. Do I have to do anything special to delete a node?

A: In C++ you have to specifically delete the node using free or delete. In java though, once nothing is pointing to the node, then it will automatically be removed.

1. Can I use any of the built-in library for linked lists?

A: No, you create everything from scratch

1. Can I read the items from the file into some other data structure, sort it and then recreate my linked list

A: No, you sort as you insert into your linked list.

1. Can you provide a general pseudocode to get me started?

//you will not be reading the entire line but rather one word at a time

Loop through the file as long as there is something to read

Read word//reads first word which could be the word delete or a name

If the string==’delete’

Read word //this means that the next word is a name

Delete (word) //look for the node and delete it

Else

//the line just contained a name which means you insert

Insert (word)

Traverse() //Traverse through the list, display in ascending, descending order

1. How many classes do I need (Only for Java programmers)

A: Node, Doubly, Driver. Both Node and Doubly will only contain instance methods and variables. Driver will contain only static methods.

Doubly contains head and tail as instance variables, delete,insert, traverseAsending, traverseDescnding methods. Since all variables are private, provide getter and setter methods

Node will contain value, prev, next as instance variables, Since all the variables will be private, provide getter and setter methods.

Driver will contain the main method which will contain the content from item 6 in this document

1. What is the code to delete a node from a doubly linked list?

A: This was discussed in the lecture

Once you find the node that you want to delete, the variable curr will point to it.if curr== null then you can skip delete all together

If (curr.prev!=null)

curr.prev.next=curr.next //a node other than head is being deleted

else

head =curr.next //the node being deleted is the head

If (curr.next!=null)

curr.next.prev=curr.prev // a node other than tail is being deleted

else

tail=curr.prev the node being deleted is the tail