CSCI 2275 – Programming and Data structures
Instructor: Hoenigman
Recitation 5

Linked List and Doubly Linked Lists

Objectives:
1. What are linked lists
2. Insertion
3. Deletion
4. Printing
5. Recitation Exercise

What is a Linked List?

A linked list is a sequence of data structures, which are connected via links.

Let's elaborate:

Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List.
1. Node − Each Node of a linked list can store data and a link.
2. Link − Each Node of a linked list contains a link to the next Node called Next. In Doubly linked list there are 2 pointers – Next and Previous, former pointing to the node after the given node and latter pointer to node before the given nodes.
3. LinkedList − A Linked List contains a connection link from the first link called Head. Every Link after the head points to another Node (unit of Linked List). The last node's Next pointer points to NULL. In doubly linked lists the previous node of head also points to NULL.

Insert in Singly Linked List:

Adding a new node in a linked list is a multi-step activity. First, we create a node using the same structure and find the location where it must be inserted.

Let us take an example, where we are inserting a node B (NewNode), between A (LeftNode) and C (RightNode).

Current Linked List:

Head(X)->Y->A->C->Z->NULL

Step 1:

Now before we add a node between A and C we need to make sure we do not loose any links i.e. Always make sure there is something pointing to all the nodes at any given point of time.

Step 2:

Now let us add a new link without changing or breaking any old links

```
B->next = C;
```

As we did not change anything we can be rest assured that none of the links were broken

Step 3:

Now let us set
A->next =B

Here the link to C was broken but we already have B->next pointing to C and we also have something pointing to B now.

Trivia: Think how insertion at head would work?
Will you have a previous node (A) ? Will you have to change head?
Similarly think of insertion at tail would work

Deletion in a singly linked list

Deletion is also a multi-step process. First, locate the target node to be removed, by using searching algorithms (and finding the required key).
You will also need the node that is before the node you have to delete. Let us assume B is the node you have to delete and A is before B and C is after B

Head(X) -> Y->A->B->C->Z->NULL

Step1:

First we need pointers to the nodes A and B. We don't need pointer to C because not the value of the pointer of C is changing.

Step2:

Set
A->next=B-next
You have now broken the link between A and B but you still have pointer to B

Step 3:

Set B->next =NULL

This step is just for safetly so that we don’t have anything extra pointing at C

Step 4:

If we need to use the deleted node, we can keep it in memory. Otherwise we can simply deallocate memory and wipe off the deleted node completely.

Think about the cases of deletion of head and tail

Note: Think about how insertion and deletion will work in a doubly linked list. You will have to change not just the Next pointers but also the previous pointer

Recitation Exercise:

Given a doubly linked list –

1<->2<->3<->5<->6->NULL

Write a function to insert a with value 4 between the nodes 3 and 5 and then print the doubly linked list

Note: Please give a huge round of applause for Sreesha who chose to conduct a very important recitation