

Implementing more linked list operations

Lecturer: Arran Stewart

Overview

- ▶ How can we insert elements into the middle of a linked list?

Introduction

- ▶ We have seen that a linked list can be used to implement a queue.
- ▶ For a queue, elements are always added at the back of the queue (the tail) and remove from the front (the head).
- ▶ But what if we want to insert elements in the middle of a linked list?

Java code for a linked list of integers

We can use the same Java code we have seen before to implement a linked list of integers:

```
public class ListNode {  
    //fields (attributes)  
    public int value;  
    public ListNode next;  
  
    //constructor  
    public ListNode(int val) {value = val; next = null; }  
}
```

ListNode explanation

- ▶ Recall that each node in the list is a `ListNode` object which contains two instance variables:
 - ▶ an `int` to be stored, called “value”, and
 - ▶ a link to the next node in the list (called “next”)

Inserting a node

If want to insert a new node, we will need to have a pointer to the node just *before* it.

- ▶ Let `newNode` be the pointer to the node to be inserted
- ▶ Let `before` be the pointer to the node already in the list.
- ▶ Set `newNode.next = before.next`, and then
- ▶ Set `before.next = newNode`

Finding a node

- ▶ Suppose we know that we want to insert the new node at some particular spot in the list; how can we write code to locate that spot?
- ▶ For instance, suppose we have a list containing the ints 1, 2, 3 and 4, and we would now like to insert a new node containing the int 99, just after the 3.
- ▶ We can start by setting our before pointer to the head of the list.
- ▶ Then we must traverse along the list, updating our before pointer, until we find we're on a node containing the number 3; at that point, we're at the correct spot in the list.

Removing a node

- ▶ Again we need a pointer to the node *before* the one we wish to delete.
- ▶ Then simply set `before.next = before.next.next`, to “splice” together the two nodes either side of the deleted node.

Removing a node

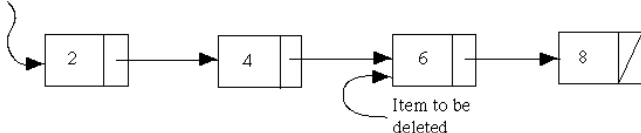
We will look at diagrams showing the process of removing a node.

We start with the original list, and need to traverse the list to get the node before the node we wish to delete. i.e. we need to find the node containing the value 4.

Original List
first



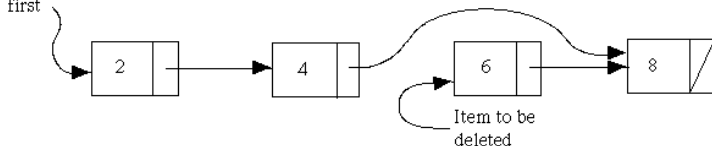
Step 1: Find item to be deleted
first



Removing a node

Once we have the node before the one to be deleted – we alter the value of its link.

Step 2: Change previous pointer
first



Step 3: Throw away old item
first

