Doubly Linked Lists For Dummy Nodes

1 Short Answers

Bring answers to the following questions to lab on Wednesday.

8.1
8.5

2 Lab

Do the laboratory at the end of Chapter 8. The goals of this lab are:

1. to practice manipulating linked data structures
2. to demonstrate how data structure design impacts boundary cases

To do this, we will take the DoublyLinkedList class from lecture and extend it in a way that greatly simplifies how we handle empty lists, and operations that affect the head and tail.

Before coming to lab, read the lab exercise and think about an implementation strategy. A good starting point is to sketch how a LinkedList will change when various operations are performed on it, as we have been doing in lecture.

The directory
/private/Network/Servers/cortland.cs.williams.edu/Volumes/Courses/cs136/labs/lab5
contains starter files. Specifically, the directory contains a minimum starter file LinkedList.java. As usual, don’t forget to run chmod u+w on the files after you copy them.

Turn in LinkedList.java and short answers to the thought questions.

Optional Extension Provide an efficient sorting algorithm for your LinkedList class. In other words, write a method

void sort(Comparator c) {
... 
}

for your class to sort the elements using a comparator. One way to do this is to use the get and set methods that provide random access to list elements. However, these operations are costly on a linked list. What would the performance of the algorithm be if you write Selection Sort using these operations? What about Quick Sort? To sort in \(O(n\log n)\) time, you will need to manipulate the list elements directly.