CS 158 Lab 7 February 4, 2009
More with Linked Lists

1. Go back into the project called Lists. Make sure the methods for **SLList** (**add**, **getSize**, **getNode**, and **remove**) all work correctly and all the tests pass. Be sure that you have "strong" tests. If you are in doubt, ask me or the assistant. Basically, your tests should make sure the methods work when you have an empty list, a list with one node, a list with many nodes, if you are removing the first node, the last node, a middle node, the only node.

2. Create another **remove** method in the **SLList** class, this one having an **Object** argument and returning a **boolean**. This method must search the list looking for a node that contains the data passed in as an argument. If such a node is found, that node must be removed, and the method returns **true**. If no such node is found, the method just returns **false**. Only one node is to be removed on any call to this method. Notice that Java has no trouble distinguishing between this method and the one you wrote earlier that takes an integer argument, since their signatures are different. You might want to write the JUnit test method for this first. Make sure you test what you need to. Hint: you do not have to "reinvent the wheel".

3. Change the **SLList** class so that it contains another protected node variable called **tail**. Modify the constructor so that it instantiates a node for **tail** to point to, and set its data field to "Tail node of singly linked list".

4. Refactor/rename the **add** method to **addHead**. Modify it so that it maintains the tail node correctly. Normally, the tail node should not change when adding a node to the list at the head. But there is one case where the tail must change: when you are adding the very first node to the list. Make the appropriate changes to the **addHead** method, and add code to the test method to make sure **addHead** changes **head** and **tail** appropriately for all relevant cases.

5. Create a new method of **SLList** called **addTail** that adds the node that is passed in as an argument to the tail end of the list. You should NOT need a loop to do this. But you will have to consider the special case where you are trying to add the very first node to the list. Add a test method that tests all the relevant cases.

6. Modify both **remove** methods so that **tail** is maintained properly. Make sure your tests contain cases for removing the only element, the first element, the last element, a middle element. Be sure that **head** and **tail** are maintained properly.

7. Add a method called **add** that takes an integer argument **n** and an **Object** argument and inserts the **Object** into a new node before the node at position **n**. Design it well, and test it thoroughly.

8. Copy the file **SLList.java** to the shared drive.