NextClass7

CS173: Intermediate Computer Science — Spring 2014

Instructor: Thomas Bressoud — 2014-02-21

In this assignment, you are tasked with drawing pictures illustrating list operations on a doubly linked list structure. In the structure, each ListNode has a *next* link reference, a *prev* link reference, and a reference to the *item* associated with the ListNode. The DLList implementation maintains a *head*, a *tail* and a *size*.
For each problem, you will draw the diagram of the situation before the operation, and then, in another color, augment the picture as it should look after the operation is completed (as we did for each operation in class). So the diagram is for a specific example. Finally, you should write down the sequence of Python statements that would accomplish the transformation, as if you were writing it as part of the method definition.
For each problem, the initial state is the same: a three-node list, named `L`, where the first node has the value 7, the second node has the value 3, and the third node has the value 9. The first two problems consider two different cases when inserting into a multi-element list, and the second two problems consider two different cases when deleting for a milt-element list. You can assume a private method, `_find(k)` that returns a reference to the ListNode at index *k*. You do not have to combine variations of insert or delete into a single unified method.

1. Insert into the middle of the list: `L.insert(1, 4)`
    - Drawing
    - Code: `def insert(self, i, x):`

2. Insert onto the end of the list: `L.insert(3, 4)`
   - Drawing
   - Code: `def insert(self, i, x):`

3. Delete from the middle of the list, returning item: `x = L._delete(1)`
   - Drawing
   - Code: `def _delete(self, i):`

4. Delete from the beginning of the list, returning item: `x = L._delete(0)`

   - Drawing

   - Code: `def _delete(self, i):`