Linked List Exercise

Code, Craft, Community

10/16/2019

1. Create a class MyLinkedList which is a singly linked, non-circular list where each node only has one link next and the list has a head and a tail link (think about how you would implement the node). The MyLinkedList class also implements the Iterable interface. NOTE: For C#, implement the IEnumerable and the IEnumerator interfaces.

The following should be implemented as well:

2. Add the methods to the MyLinkedList class:

    a. iterator() to implement the Iterable interface. This method returns an instance of MyLinkedListIterator initialized appropriately.

    b. addFirst(<E> value) that adds a value to the front of the list

    c. addEnd(<E> value) that adds a value to the end of the list - Warning: remember to deal with the boundary cases!!!

    d. A get method which given an index returns the element.

3. Create a class MyLinkedListIterator that implements the Iterator interface.

4. Similarly to Lab 2, create a class called MyListStringContainer which uses your new data structure. It only needs to implement the following methods:

    a. addToBack

    b. addToFront

    c. Search for a substring (this should search for a String as a substring of a String in the list and return the first index of such a String if it is present, if not return -1. In this case do not assume that the array is sorted). You should write two versions of this method. One which uses an iterator and one which does not.

5. Write an ExperimentController so that it now evaluates the performance of both searches against each other. As usual, summarize the results of your experiments in your report, including appropriate graphs.

6. Unit test your classes.