Ian Kaiserman

4/10/2020

CSCD 300-002

Homework 1 Report

Homework 1 Report

This is a pseudo-code-style report on all 9 methods being added in Homework 1

I. Public Object removeFirst() throws Exception
    1. Create Node reference cur assigned to the next reference of head
    2. Reassign the next reference of head to the next reference of cur
    3. Decrease size by 1
    4. Return the data object of Node cur (removed Node)

II. Public Boolean contains(Object o)
    1. Create Node reference cur assigned to the next reference of head
    2. Create a while loop with the looping condition cur is not null
        i. If statement checking if the data reference of cur equals the object passed in, or if they are both null
            1. If the data is equal to the object, or they are both null, return true
        ii. Reassign cur to next reference of cur (iteration)
    3. If the method has not returned true, meaning data was not found, return false

III. Public Boolean remove(Object o)
    1. Create Node references cur assigned to the next reference of head, and prev assigned to head itself
    2. Create a while loop with the looping condition cur is not null

        i. If statement checking if the data reference of cur equals the object passed in, or if they are both null

            1. If the data is equal to the object, or if they are both null, reassign the next reference of prev to the next reference of cur
            2. Decrease size by 1
            3. Return true

        ii. Reassign prev to cur, and cur to the next reference of cur (iteration)

    3. If the method did not return true, meaning passed in data was not found, return false

IV. Public Boolean removeAllCopies(Object o)

    1. Create Node references cur assigned to the next reference of head, and prev assigned to head itself
    2. Create boolean success equal to false which will be used later
    3. Create a while loop with the looping condition cur is not null

        i. If statement checking if the data reference of cur equals the object passed in, or if they are both null

            1. If the data is equal to the object, or if they are both null, reassign the next reference of prev to the next reference of cur
            2. Decrease size by 1
            3. Reassign success to true
            4. Else, reassign prev to cur (to fix consecutive removal)

        ii. Reassign cur to the next reference of cur

    4. Return success (returning false means the data was never found)

V. Public static MyLinkedList interleave(MyLinkedList A, MyLinkedList B)

1. If statement checking if A is empty
    i. If true, return B
2. If statement checking if B is empty
    i. If true, return A
3. Create empty MyLinkedList object called result
4. Create Node references cur1 assigned to the next reference of the head of A, and cur2 assigned to the next reference of the head of B
5. Create while loop with looping condition cur1 is not null
    i. Have result call add method (defined later) passing in data reference of cur1
    ii. Increase size of result
    iii. If statement checking if cur2 is not null (this will trigger until reaching the end of B, edge case for B being shorter than A)
        1. If true, have result call add method passing in data reference of cur2
        2. Increase the size of result
    iv. Reassign cur1 to next reference of cur1, and cur2 to next reference of cur2 (iteration of both lists)
6. After exiting while loop, create a new while loop with looping condition cur2 is not null (edge case for B being longer than A)
    i. Have result call add method passing in data reference of cur2
    ii. Increase the size of result

7. Return result

VI. Public void add(int index, Object o) {

1. Create Node references cur assigned to the next reference of head, and prev assigned to head itself
2. Create Node nn with parameters o (passed in) and null
3. For loop starting at 0, looping condition i is less than index (passed in)
    i. Reassign prev to cur, and cur to the next reference of cur
4. Cur is now located at the index to be added. Reassign the next reference of nn to cur, and the next reference of prev to the new node nn
5. Increase size by 1

VII. Public Object get(int index)

1. Create node reference cur assigned to the next reference of head
2. For loop starting at 0, looping condition i is less than index (passed in)
    i. Reassign cur to the next reference of cur
3. Cur is now located at the index to check. Return the data of cur

VIII. Public Object remove(int index)

1. Create node references cur assigned to the next reference of head, and prev assigned to head itself
2. For loop starting at 0, looping condition i is less than index (passed in)
    i. Reassign prev to cur, and cur to the next reference of cur
3. Cur is now located at the index to remove. Reassign the next reference of prev to the next reference of cur
4. Decrease size by 1

5. Return the data of cur

IX. Public Boolean add(Object e)

1. Create Node reference cur assigned to head
2. Create new Node nn with parameters e (passed in) and null
3. While loop with looping condition next reference of cur is not null
    i. Reassign cur to the next reference of cur
4. Cur is now located at the very last Node that has data. Reassign the next reference of cur to nn
5. Increase size by 1