CS 1083

Assignment #9

Author: Yulong Wang

Id: 3713596

1. Source Code:

IntList.java

```java
/**
Defines a class that represents a list of integers
*/
public class IntList {

    /**
    First node in the list
    */
    private IntNode front;

    /**
     * Constructs a list.  Initially the list is empty.
     */
    public IntList() {
        front = null;
    }

    /**
     * Adds given integer to front of list.
     *
     * @param val integer to add to the front of the list
     */
    public void addToFront(int val) {
        front = new IntNode(val, front);
    }

    /**
     * Removes the first node from the list.
     * If the list is empty, does nothing.
     */
    public void removeFirst() {
        if (front != null) {
            front = front.next;
        }
    }

    /**
     * Prints the list elements from first to last.
     */
    public void print() {
        System.out.println("--------------------");
        System.out.print("List elements: ");
        IntNode temp = front;
        while (temp != null) {
            System.out.print(temp.val + " ");
            temp = temp.next;
        }
```

```
        System.out.println("\n-----------------------\n");
    }

    /**
     * return the length of the linked list
     * @return int the length of the linked list
     */
    public int length(){
        IntNode temp = front;
        int length = 0;
        while(temp!=null){
            temp = temp.next;
            length++;
        }
        return length;
    }

    /**
     * add a node to the end of the list
     * @param val the val to be added
     */
    public void addToEnd(int val){
        if(front!=null) {
            IntNode temp = front;
            while (temp.next != null) {
                temp = temp.next;
            }
            temp.next = new IntNode(val, null);
        }else{
            front = new IntNode(val, null);
        }
    }

    /**
     * remove the last element in the linked list
     */
    public void removeLast(){
        if(front!=null){
            if(front.next==null){
                front = null;
                return;
            }
            IntNode temp = front;
            IntNode pre = temp;
            while(temp.next!=null){
                pre = temp;
                temp = temp.next;
            }
            pre.next = null;
        }
    }
```

```java
    /**
     * replace all oldVal with newVal in the linked list
     * @param oldVal old int value
     * @param newVal new int value
     */
    public void replace(int oldVal, int newVal){
        IntNode temp = front;
        while(temp!=null){
            if(temp.val == oldVal){
                temp.val = newVal;
            }
            temp = temp.next;
        }
    }

    /**
     * print the link recursively
     */
    public void printRec(){
        recHelper(front);
    }

    /**
     * helper function
     * @param temp
     */
    private void recHelper(IntNode temp){
        if(temp==null){
            return;
        }
        System.out.println(temp.val);
        recHelper(temp.next);
    }

    /**
     * print linked list backwards recursivly
     */
    public void printRecBackwards(){
        recBackwardsHelper(front);
    }

    /**
     * helper function
     * @param temp
     */
    private void recBackwardsHelper(IntNode temp){
        if(temp==null){
            return;
        }
        recBackwardsHelper(temp.next);
        System.out.println(temp.val);
    }
```

```
    /**
    An inner class that represents a node in the integer list.
    The public variables are accessed by the IntList class.
    */
    private class IntNode {

        /**
        Value stored in node.
        */
        public int val;

        /**
        Link to next node in list.
        */
        public IntNode next;

        /**
        Constructor; sets up the node given a value and IntNode reference
        @param val the value to store in the node
        @param next the link to the next node in the list
        */
        public IntNode(int val, IntNode next) {
            this.val = val;
            this.next = next;
        }
    }
}
```

IntListTest.java

```
import java.util.Scanner;

/**
Tests the methods of the IntList class.
*/
public class IntListTest {
    private static Scanner scan;
    private static IntList list = new IntList();

    /*
    Creates a list, then repeatedly prints the menu and does what
    the user asks until they quit.
    */
    public static void main(String[] args) {
        scan = new Scanner(System.in);
        printMenu();
        int choice = scan.nextInt();
        while (choice != 0) {
            dispatch(choice);
            printMenu();
            choice = scan.nextInt();
```

```
    }
}

/*
Does what the menu item calls for.
*/
public static void dispatch(int choice) {
    int newVal;
    switch (choice) {
        case 0:
            System.out.println("Bye!");
            break;
        case 1: //add to front
             System.out.println("Enter integer to add to front");
             newVal = scan.nextInt();
             list.addToFront(newVal);
             break;
        case 2: //remove first element
             list.removeFirst();
             break;
        case 3: //print
             list.print();
             break;
        case 4: //length
             System.out.println(list.length());
             break;
        case 5: //addToEnd
             System.out.println("Enter integer to add to end");
             newVal = scan.nextInt();
             list.addToEnd(newVal);
             break;
        case 6: //removeLast
             list.removeLast();
             break;
        case 7: //replace
             System.out.println("Enter old integer to be replaced");
             int oldVal = scan.nextInt();
             System.out.println("Enter new integer to replace with");
             newVal = scan.nextInt();
             list.replace(oldVal,newVal);
             break;
        case 8: // print rec
             list.printRec();
             break;
        case 9: // print rec
             list.printRecBackwards();
             break;
        default:
            System.out.println("Sorry, invalid choice");
    }
}
```

```
    /*
    Prints the user's choices
    */
    public static void printMenu() {
        System.out.println("\n Menu ");
        System.out.println(" ====");
        System.out.println("0: Quit");
        System.out.println("1: Add an integer to the front of the list");
        System.out.println("2: Remove an integer from the front of the list");
        System.out.println("3: Print the list");
        System.out.println("4: returns the number of elements in the list");
        System.out.println("5: takes an integer and puts it on the end of the list");
        System.out.println("6: removes the last element of the list. If the list is
empty, does nothing.");
        System.out.println("7: replaces all occurrences of oldVal in the list with
newVal.");
        System.out.println("8: Print the list recursively");
        System.out.println("9: Print the list backwards recursively");
        System.out.print("\nEnter your choice: ");
    }
}
```

2. Test

Note: I manually delete duplicated menus to make the report shorter.

```
 Menu
 ====
0: Quit
1: Add an integer to the front of the list
2: Remove an integer from the front of the list
3: Print the list
4: returns the number of elements in the list
5: takes an integer and puts it on the end of the list
6: removes the last element of the list. If the list is empty, does nothing.
7: replaces all occurrences of oldVal in the list with newVal.
8: Print the list recursively
9: Print the list backwards recursively

Enter your choice: 4
0

Enter your choice: 6

Enter your choice: 7
Enter old integer to be replaced
1
Enter new integer to replace with
1

Enter your choice: 3
```

```
--------------------
List elements:
----------------------


Enter your choice: 8

Enter your choice: 9

Enter your choice: 3
--------------------
List elements:
----------------------

Enter your choice: 1
Enter integer to add to front
1

Enter your choice: 5
Enter integer to add to end
2

Enter your choice: 5
Enter integer to add to end
3


Enter your choice: 4
3


Enter your choice: 7
Enter old integer to be replaced
1
Enter new integer to replace with
0

Enter your choice: 3
--------------------
List elements: 0 2 3
----------------------


Enter your choice: 8
0
2
3

Enter your choice: 9
3
2
0
```

```
Enter your choice: 6

Enter your choice: 6

Enter your choice: 6

Enter your choice: 3
--------------------
List elements:
----------------------

Enter your choice: 6

Enter your choice: 5
Enter integer to add to end
1

Enter your choice: 3
--------------------
List elements: 1
----------------------


Enter your choice: 8
1

Enter your choice: 9
1


Enter your choice: 7
Enter old integer to be replaced
1
Enter new integer to replace with
0


Enter your choice: 3
--------------------
List elements: 0
----------------------
```