Solution to CSE143 Section #6 Problems

```
// Class LinkedIntList can be used to store a list of integers.

public class LinkedIntList {
    private ListNode front;  // first value in the list

    // post: constructs an empty list
    public LinkedIntList() {
        front = null;
    }

    // post: returns the current number of elements in the list
    public int size() {
        int count = 0;
        ListNode current = front;
        while (current != null) {
            current = current.next;
            count++;
        }
        return count;
    }

    // pre : 0 <= index < size()
    // post: returns the integer at the given index in the list
    public int get(int index) {
        ListNode current = front;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        return current.data;
    }

    // post: creates a comma-separated, bracketed version of the list
    public String toString() {
        if (front == null) {
            return "[]";
        } else {
            String result = "[" + front.data;
            ListNode current = front.next;
            while (current != null) {
                result += ", " + current.data;
                current = current.next;
            }
            result += "]";
            return result;
        }
    }
```

```
// post : returns the position of the first occurrence of the given
//        value (-1 if not found)
public int indexOf(int value) {
    int index = 0;
    ListNode current = front;
    while (current !=  null) {
        if (current.data == value) {
            return index;
        }
        index++;
        current = current.next;
    }
    return -1;
}

// post: appends the given value to the end of the list
public void add(int value) {
    if (front == null) {
        front = new ListNode(value);
    } else {
        ListNode current = front;
        while (current.next != null) {
            current = current.next;
        }
        current.next = new ListNode(value);
    }
}

// pre: 0 <= index <= size()
// post: inserts the given value at the given index
public void add(int index, int value) {
    if (index == 0) {
        front = new ListNode(value, front);
    } else {
        ListNode current = front;
        for (int i = 0; i < index - 1; i++) {
            current = current.next;
        }
        current.next = new ListNode(value, current.next);
    }
}

// pre : 0 <= index < size()
// post: removes value at the given index
public void remove(int index) {
    if (index == 0) {
        front = front.next;
    } else {
        ListNode current = front;
        for (int i = 0; i < index - 1; i++) {
```

```
                current = current.next;
            }
            current.next = current.next.next;
        }
    }
}
```