Discussion Section 3 (Thomas Feng, 09/11/2008)

Many problems can be solved using either a recursive or iterative solution. While both are correct, the best one is determined by other factors, such as the size of the input, the ease of maintaining the code, and the system on which the code runs. In this exercise, we'll ask you to write down both a recursive and iterative solution. In both cases, the program should insert and integer into a sorted IntList in the correct position, i.e. so the list remains sorted after the insertion. We're only asking for the destructive solution, but you should think about how your code changes for the non-destructive case. You're encouraged to work in small groups (2-3 people).

1. Show a recursive function that inserts an integer into a sorted IntList such that the list is still sorted after the insertion.

```
static IntList dInsertSortedRecursive(IntList L,
        int value){
    /* Your code here. */









}
```

2. Show a iterative function that inserts an integer into a sorted IntList such that the list is still sorted after the insertion.

```
static IntList dInsertSortedIterative(IntList L,
        int value){
    /* Your code here. */









}
```

Discussion Section 3 Solution:

```
public class IntListSort {

    static IntList dInsertSortedRecursive(IntList L, int value){
        if(L == null || value < L.head)
            L = new IntList(value, L);
        else
            L.tail = dInsertSortedRecursive(L.tail, value);
        return L;
    }

    static IntList dInsertSortedIterative(IntList L, int value){
        if(L == null || value < L.head)
            return new IntList(value, L);
        else {
            IntList c = L;
            while (c.tail != null && c.tail.head <= value) {
                c = c.tail;
            }
            c.tail = new IntList(value, c.tail);
            return L;
        }
    }

    static void print(IntList L) {
        System.out.print("[");
        while (L != null) {
            System.out.print(L.head);
            if (L.tail != null) {
                System.out.print(", ");
            }
            L = L.tail;
        }
        System.out.println("]");
    }

    public static void main(String[] args) {
        print(dInsertSortedIterative(IntList.list(1, 2, 4, 5), 3));
    }
}
```