

**CHANDIGARH UNIVERSITY
UNIVERSITY INSTITUTE OF NGINEERING
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



Submitted By: Vivek Kumar(21BCS8129)		Submitted To: Mamta Punia(E12337)	
Subject Name	Competitive Coding - I		
Subject Code	20CSP-314		
Branch	Computer Science and Engineering		
Semester	5 th		

Experiment No. - 3

Student Name: Vivek Kumar

Branch: BE-CSE(LEET)

Semester: 5th

Subject Name: Competitive coding - I

UID: 21BCS8129

Section/Group: WM-20BCS-616/A

Date of Performance: 02/09/2022

Subject Code: 20CSP-314

Compare two Linked list:

1. Aim/Overview of the practical:

You're given the pointer to the head nodes of two linked lists. Compare the data in the nodes of the linked lists to check if they are equal. If all data attributes are equal and the lists are the same length, return 0. Otherwise, return 1.

2. Task to be done/ Which logistics used:

Example

$l1ist1 = 1 \rightarrow 2 \rightarrow 3 \rightarrow NULL$

$l1ist2 = 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow NULL$

The two lists have equal data attributes for the first 3 nodes. $l1ist2$ is longer, though, so the lists are not equal. Return 0.

Function Description

Complete the `compare_lists` function in the editor below.

`compare_lists` has the following parameters:

- `SinglyLinkedListNode l1ist1`: a reference to the head of a list
- `SinglyLinkedListNode l1ist2`: a reference to the head of a list

Returns

- `int`: return 1 if the lists are equal, or 0 otherwise

Input Format

The first line contains an integer t , the number of test cases.

Each of the test cases has the following format:

The first line contains an integer n , the number of nodes in the first linked list.

Each of the next n lines contains an integer, each a value for a data attribute.

The next line contains an integer m , the number of nodes in the second linked list.

Each of the next m lines contains an integer, each a value for a data attribute.

Constraints

- $1 \leq t \leq 10$
- $1 \leq n, m \leq 1000$
- $1 \leq l1ist1[i], l1ist2[i] \leq 1000$

Output Format

Compare the two linked lists and return 1 if the lists are equal. Otherwise, return 0. Do NOT print anything to stdout/console.

The output is handled by the code in the editor and it is as follows:

For each test case, in a new line, print 1 if the two lists are equal, else print 0.

Sample Input

```
2
2
1
2
1
1
2
1
2
2
1
2
```

Sample Output

```
0
1
```

Explanation

There are $t = 2$ test cases, each with a pair of linked lists.

- In the first case, linked lists are: 1 -> 2 -> NULL and 1 -> NULL
- In the second case, linked lists are: 1 -> 2 -> NULL and 1 -> 2 -> NULL

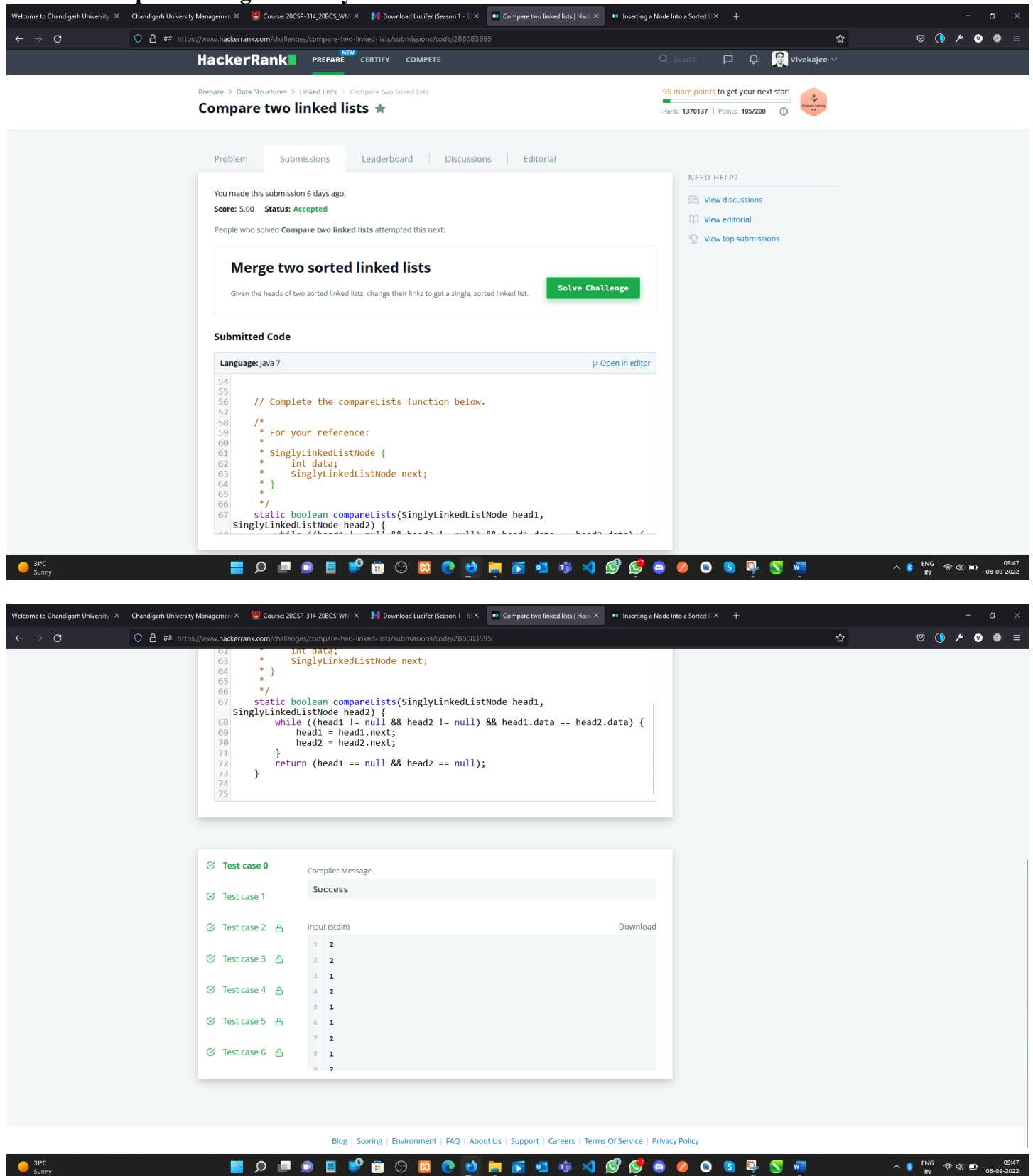
3. Hardware and Software Requirements (For programming-based labs):

- Laptop or Desktop
- Hacker-Rank Account

4. Steps for experiment/practical/Code:

```
static boolean compareLists(SinglyLinkedListNode head1, SinglyLinkedListNode head2) {
    while ((head1 != null && head2 != null) && head1.data == head2.data) {
        head1 = head1.next;
        head2 = head2.next;
    }
    return (head1 == null && head2 == null);
}
```

5. Result/Output/Writing Summary:



Welcome to Chandigarh University | Chandigarh University Management | Course: 20CSP-314_20BCS_WM | Download Lucifer (Season 1 - 6) | Compare two linked lists | Hack | Inserting a Node Into a Sorted | +

https://www.hackerrank.com/challenges/compare-two-linked-lists/submissions/code/288083695

HackerRank PREPARE NEW CERTIFY COMPETE

Prepare > Data Structures > Linked Lists > Compare two linked lists

Compare two linked lists ★

95 more points to get your next star! Rank: 1370137 | Points: 105/200

Problem Submissions Leaderboard Discussions Editorial

You made this submission 6 days ago.
Score: 5.00 Status: Accepted

People who solved Compare two linked lists attempted this next:

Merge two sorted linked lists
Given the heads of two sorted linked lists, change their links to get a single, sorted linked list. [Solve Challenge](#)

NEED HELP?
[View discussions](#)
[View editorial](#)
[View top submissions](#)

Submitted Code

Language: java 7 [Open in editor](#)

```

54
55
56 // Complete the compareLists function below.
57
58 /*
59  * For your reference:
60  * SinglyLinkedListNode {
61  *   int data;
62  *   SinglyLinkedListNode next;
63  * }
64  */
65
66
67 static boolean compareLists(SinglyLinkedListNode head1,
68 SinglyLinkedListNode head2) {
69     while ((head1 != null && head2 != null) && head1.data == head2.data) {
70         head1 = head1.next;
71         head2 = head2.next;
72     }
73     return (head1 == null && head2 == null);
74 }
75

```

Test case 0 Success

Test case 1 Success

Test case 2 Success

Test case 3 Success

Test case 4 Success

Test case 5 Success

Test case 6 Success

Compiler Message

Success

Input (stdin) Download

```

1 2
2 2
3 1
4 2
5 1
6 1
7 2
8 1
9 2

```

[Blog](#) | [Scoring](#) | [Environment](#) | [FAQ](#) | [About Us](#) | [Support](#) | [Careers](#) | [Terms Of Service](#) | [Privacy Policy](#)

Inserting a Node Into a Sorted Doubly Linked List:

1. Aim/Overview of the practical:

Given a reference to the head of a doubly-linked list and an integer, data, create a new *DoublyLinkedListNode* object having data value data and insert it at the proper location to maintain the sort.

2. Task to be done/ Which logistics used:

Example

head refers to the list $1 \leftrightarrow 2 \leftrightarrow 4 \rightarrow NULL$

data = 3

Return a reference to the new list: $1 \leftrightarrow 2 \leftrightarrow 3 \leftrightarrow 4 \rightarrow NULL$.

Function Description

Complete the `sortedInsert` function in the editor below.

`sortedInsert` has two parameters:

- `DoublyLinkedListNode` pointer *head*: a reference to the head of a doubly-linked list
- `int data`: An integer denoting the value of the *data* field for the `DoublyLinkedListNode` you must insert into the list.

Returns

- `DoublyLinkedListNode` pointer: a reference to the head of the list

Note: Recall that an empty list (i.e., where *head* = `NULL`) and a list with one element are sorted lists.

Input Format

The first line contains an integer *t*, the number of test cases.

Each of the test case is in the following format:

- The first line contains an integer *n*, the number of elements in the linked list.
- Each of the next *n* lines contains an integer, the data for each node of the linked list.
- The last line contains an integer, *data*, which needs to be inserted into the sorted doubly-linked list.

Constraints

- $1 \leq t \leq 10$
- $1 \leq n \leq 1000$
- $1 \leq \text{DoublyLinkedListNode.data} \leq 1000$

Sample Input

```
STDIN  Function
-----
1      t = 1
4      n = 4
1      node data values = 1, 3, 4, 10
3
4
10
5      data = 5
```

Sample Output

```
1 3 4 5 10
```

Explanation

The initial doubly linked list is: $1 \leftrightarrow 3 \leftrightarrow 4 \leftrightarrow 10 \rightarrow NULL$.

The doubly linked list after insertion is: $1 \leftrightarrow 3 \leftrightarrow 4 \leftrightarrow 5 \leftrightarrow 10 \rightarrow NULL$

3. Hardware and Software Requirements (For programming-based labs):

- Laptop or Desktop
- Hacker-Rank Account

4. Steps for experiment/practical/Code:

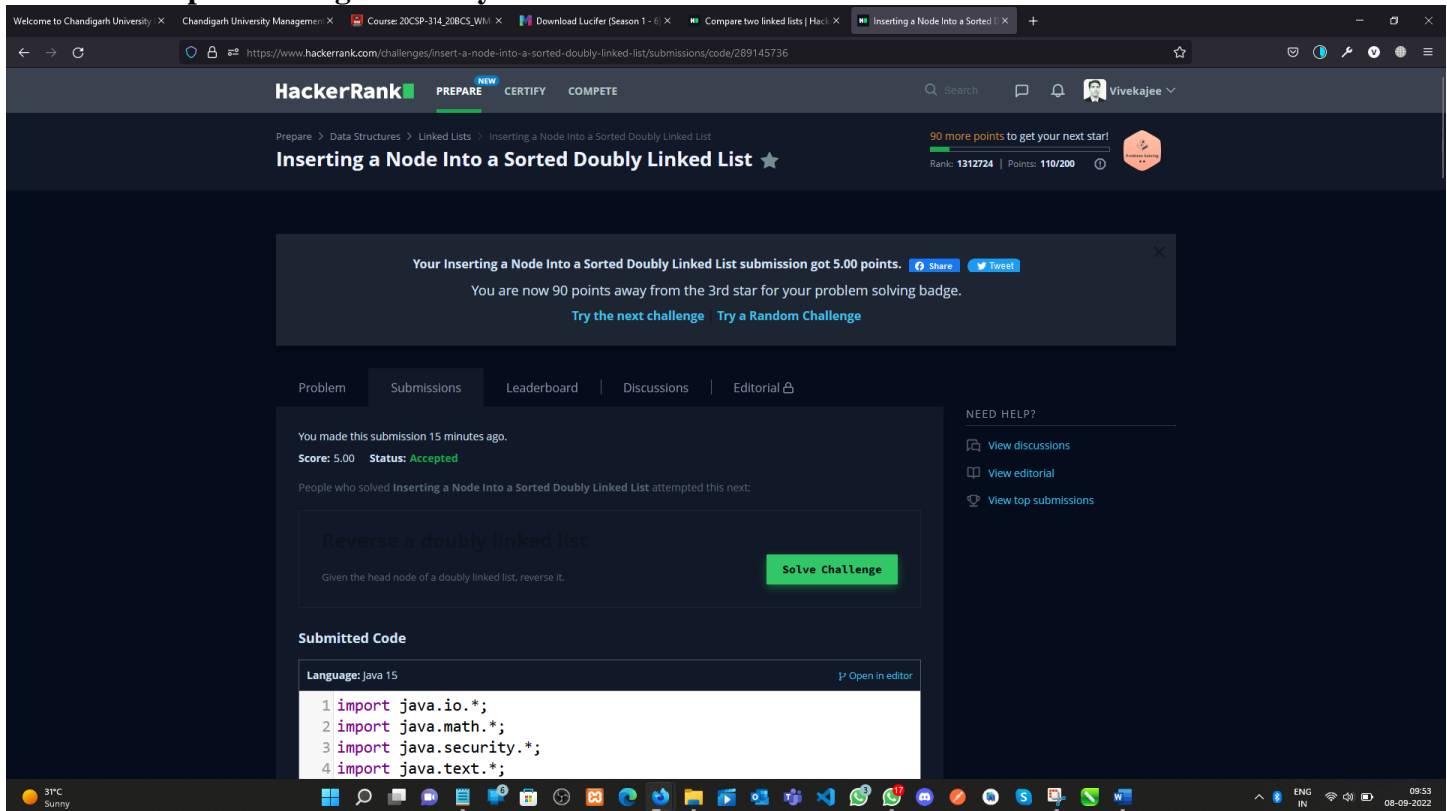
```
public static DoublyLinkedListNode sortedInsert(DoublyLinkedListNode llist, int data) {
    // Write your code here
    DoublyLinkedListNode node = new DoublyLinkedListNode(data);
    DoublyLinkedListNode temp = llist;
    if(llist.data >= node.data){
        node.next = llist;
        llist.prev = node;
        llist = node;
        return llist;
    }
    while(temp != null ){
        if(node.data < temp.data){
            DoublyLinkedListNode early = temp.prev;
            node.next = temp;
            node.prev = temp.prev;
            temp.prev = node;
        }
        temp = temp.next;
    }
}
```

```

        early.next = node;
        return llist;
    }
    if(temp.next == null){
        temp.next = node;
        node.prev = temp;
        return llist;
    }
    temp = temp.next;
}
return llist;
}

```

6. Result/Output/Writing Summary:



The screenshot shows the HackerRank interface for the problem "Inserting a Node Into a Sorted Doubly Linked List". The submission status is "Accepted" with a score of 5.00. A notification banner at the top states: "Your Inserting a Node Into a Sorted Doubly Linked List submission got 5.00 points. You are now 90 points away from the 3rd star for your problem solving badge." Below the notification, there are tabs for "Problem", "Submissions", "Leaderboard", "Discussions", and "Editorial". The "Submissions" tab is active, showing the submitted code in Java 15. The code is as follows:

```

1 import java.io.*;
2 import java.math.*;
3 import java.security.*;
4 import java.text.*;

```

On the right side, there is a "NEED HELP?" section with links to "View discussions", "View editorial", and "View top submissions".

Welcome to Chandigarh University | Chandigarh University Management | Course: 20CSP-314_20BCS_WM | Download Lucifer (Season 1 - 6) | Compare two linked lists | Hack | Inserting a Node Into a Sorted | +

<https://www.hackerrank.com/challenges/insert-a-node-into-a-sorted-doubly-linked-list/submissions/code/289145736>

```

83
84     public static DoublyLinkedListNode
sortedInsert(DoublyLinkedListNode llist, int data) {
85         // Write your code here
86         DoublyLinkedListNode node = new
DoublyLinkedListNode(data);
87         DoublyLinkedListNode temp = llist;
88         if(llist.data>=node.data){
89             node.next = llist;
90             llist.prev = node;
91             llist = node;
92             return llist;
93         }
94         while(temp != null ){
95             if(node.data < temp.data){
96                 DoublyLinkedListNode early = temp.prev;
97                 node.next = temp;
98                 node.prev = temp.prev;
99                 temp.prev = node;
100                early.next = node;
101                return llist;
102            }
103            if(temp.next == null){
104                temp.next = node;
105                node.prev = temp;
106                return llist;
107            }
108            temp = temp.next;
109        }
110        return llist;
111    }
112 }
113
114

```

```

144         } catch (IOException ex) {
145             throw new RuntimeException(ex);
146         }
147     });
148
149     bufferedReader.close();
150     bufferedWriter.close();
151 }
152 }
153

```

Test case 0
Test case 1
Test case 2
Test case 3
Test case 4
Test case 5
Test case 6

Compiler Message
Success

Input (stdin)
Download

```

1 1
2 4
3 1
4 3
5 4
6 10
7 5

```

Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy



Learning outcomes (What I have learnt):

1. Concept of LinkedList & Doubly LinkedList.
2. Completed my two questions.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			