**Flattening a Linked List: -**

Given a Linked List of size N, where every node represents a linked list and contains two pointers of its type:  
(i) a**next**pointer to the next node,  
(ii) a**bottom** pointer to a linked list where this node is head.

**Note:**

The **flattened**list will be printed using the **bottom**pointer **instead**of **next**pointer.

List is sorted horizontally and vertically.

**Example 1:**

**Input:**

5 -> 10 -> 19 -> 28

| | | |

7 20 22 35

| | |

8 50 40

| |

30 45

**Output:**  5-> 7-> 8- > 10 -> 19-> 20->

22-> 28-> 30-> 35-> 40-> 45-> 50.

**Note:** | represents the bottom pointer.

**Your Task:**  
You need to complete the function **flatten()** that takes **head**of the list as **parameter**and **returns**the **root**of **flattened**list. The printing is done by the **driver code**.

**Note**: Try to solve the problem without using any extra space.

**Expected Time Complexity:** O(N\*M)  
**Expected Auxiliary Space:** O(1)

**Constraints:**  
0 <= N <= 50  
1 <=**Mi**<= 20  
1 <= Element of linked list <= 103