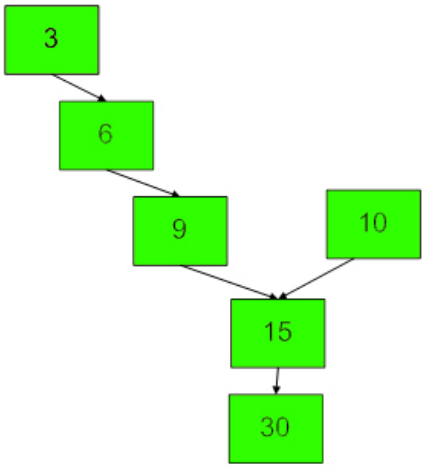
Section No.	01	Section Name	Coding for Product Development Companies
Q Paper No.	02	Topic Name	Data Structure - Linked Lists
Total Marks	30	Time Limit	90 minutes

Q.1) Intersection Point in Y Shaped Linked Lists

There are two singly linked lists in a system. By some programming error the end node of one of the linked list got linked into the second list, forming an inverted **Y** shaped list. Write a program to get the point where two linked lists merge.



Given diagram shows an example with two linked list having 15 as intersection point.
Expected time complexity is O(m + n) where m and n are lengths of two linked lists

Input:
You have to complete the method which takes two arguments as heads of two linked lists.

Output:
The function should return data value of a node where two linked lists merge. If linked list do not merge at any point, then it should return -1.

Constraints:
1 <=T<= 50
1 <=N<= 100
1 <=value<= 1000

Example:
Input:
2
2 3 2
10 20
30 40 50
5 10
2 3 0
10 20
30 40 50
First line is number of test cases. Every test case has four lines. First line of every test case contains three numbers, x (number of nodes before merge point in 1st list), y(number of nodes before merge point in 11nd list) and z (number of nodes after merge point). Next three lines contain x, y and z values.
Output:
5
-1
Note: The **Input/ Output** format and **Example** given are used for system's internal purpose, and should be used by a user for **Expected Output** only. As it is a function problem, hence a user should not read any input from stdin/console, and should not print anything on stdout/console. The task is to complete the function specified, and not to write the full code.

Q.2) Delete Middle of Linked List

Given a singly linked list, delete middle of the linked list. For example, if given linked list is 1->2->3->4->5 then linked list should be modified to 1->2->4->5

If there are even nodes, then there would be two middle nodes, we need to delete the second middle element. For example, if given linked list is 1->2->3->4->5->6 then it should be modified to 1->2->3->5->6.

If the input linked list is NULL, then it should remain NULL.

If the input linked list has 1 node, then this node should be deleted and new head should be returned.

Input:
You have to complete the method which takes one argument: the head of the linked list. You should not read any input from stdin/console.
The struct Node has a data part which stores the data and a next pointer which points to the next element of the linked list.
There are multiple test cases. For each test case, this method will be called individually.

Output:
Your function should return head of the modified linked list. If linked list is empty then it should return NULL.

Constraints:
1 <=T<= 50
1 <=N<= 1000
1 <=value<= 1000

Example:
Input:
2
5
1 2 3 4 5
6
2 4 6 7 5 1

Output:
1 2 4 5
2 4 6 5 1

Note: The **Input/Ouput** format and **Example** given are used for system's internal purpose, and should be used by a user for **Expected Output** only. As it is a function problem, hence a user should not read any input from stdin/console, and should not print anything on stdout/console. The task is to complete the function specified, and not to write the full code.

Q.3) Reorder List

Given a singly linked list: $A_0 \rightarrow A_1 \rightarrow \ldots \rightarrow A_{n-1} \rightarrow A_n$,
reorder it to: $A_0 \rightarrow A_n \rightarrow A_1 \rightarrow A_{n-1} \rightarrow A_2 \rightarrow A_{n-2} \rightarrow \ldots$
Given 1->2->3->4->5 its reorder is 1->5->2->4->3.
It is recommended do this in-place without altering the nodes' values.

Input:
In this problem, method takes one argument: Address of the head of the linked list. The function should not read any input from stdin/console.
The node structure has a data part which stores the data and a next pointer which points to the next element of the linked list.
There are multiple test cases. For each test case, this method will be called individually.

Output:
Reorder it as explained above.

Constraints:
1<=T<=200
1<=N<=200

Example:
Input:
2
3
1 2 3
4
1 7 3 4

Output:
1 3 2
1 4 7 3