This challenge is part of a tutorial track by MyCodeSchool

You’re given the pointer to the head node of a doubly linked list. Reverse the order of the nodes in the list. The head node might be NULL to indicate that the list is empty. Change the *next* and *prev* pointers of all the nodes so that the direction of the list is reversed. Return a reference to the head node of the reversed list.

Function Description

Complete the *reverse* function in the editor below. It should return a reference to the head of your reversed list.

reverse has the following parameter(s):

- *head*: a reference to the head of a DoublyLinkedList

Input Format

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

Each test case is of the following format:

- The first line contains an integer n, the number of elements in the linked list.
- The next n lines contain an integer each denoting an element of the linked list.

Constraints

- $1 \leq t \leq 10$
- $0 \leq n \leq 1000$
- $0 \leq DoublyLinkedListNode.data \leq 1000$

Output Format

Return a reference to the head of your reversed list. The provided code will print the reverse array as a one line of space-separated integers for each test case.

Sample Input

```
1
4
1
2
3
4
```

Sample Output

```
4 3 2 1
```

Explanation

The initial doubly linked list is: $1 \leftrightarrow 2 \leftrightarrow 3 \leftrightarrow 4 \rightarrow NULL$

The reversed doubly linked list is: $4 \leftrightarrow 3 \leftrightarrow 2 \leftrightarrow 1 \rightarrow NULL$