

Reverse a linked list

This challenge is part of a tutorial track by [MyCodeSchool](#) and is accompanied by a video lesson.

Given the pointer to the head node of a linked list, change the `next` pointers of the nodes so that their order is reversed. The head pointer given may be null meaning that the initial list is empty.

Example

head references the list $1 \rightarrow 2 \rightarrow 3 \rightarrow \text{NULL}$

Manipulate the *next* pointers of each node in place and return *head*, now referencing the head of the list $3 \rightarrow 2 \rightarrow 1 \rightarrow \text{NULL}$.

Function Description

Complete the *reverse* function in the editor below.

reverse has the following parameter:

- *SinglyLinkedListNode pointer head*: a reference to the head of a list

Returns

- *SinglyLinkedListNode pointer*: a reference to the head of the reversed list

Input Format

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

Each test case has 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* values of the elements in the linked list.

Constraints

- $1 \leq t \leq 10$
- $1 \leq n \leq 1000$
- $1 \leq \text{list}[i] \leq 1000$, where *list*[*i*] is the *i*th element in the list.

Sample Input

```
1
5
1
2
3
4
5
```

Sample Output

5 4 3 2 1

Explanation

The initial linked list is: **1** → **2** → **3** → **4** → **5** → *NULL*.

The reversed linked list is: **5** → **4** → **3** → **2** → **1** → *NULL*.