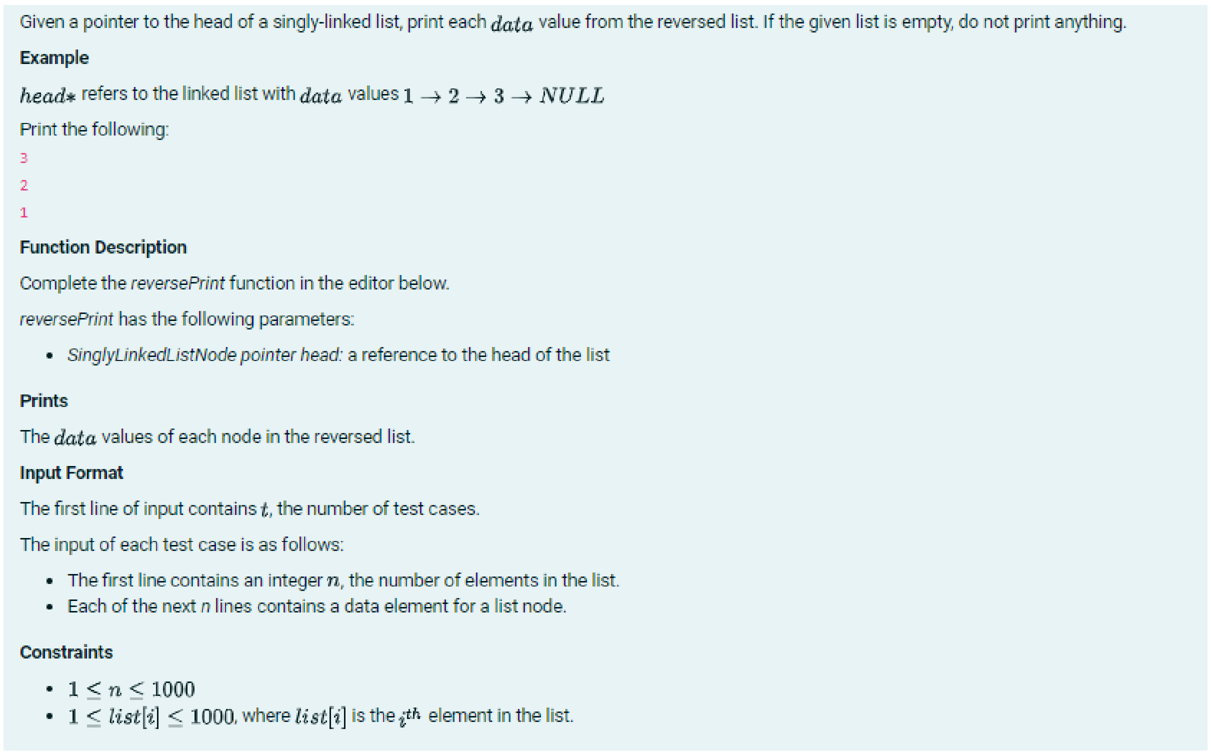
1.



Given a pointer to the head of a singly-linked list, print each $data$ value from the reversed list. If the given list is empty, do not print anything.

Example

$head*$ refers to the linked list with $data$ values $1 \rightarrow 2 \rightarrow 3 \rightarrow NULL$

Print the following:

```
3
2
1
```

Function Description

Complete the *reversePrint* function in the editor below.

reversePrint has the following parameters:

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

Prints

The $data$ values of each node in the reversed list.

Input Format

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

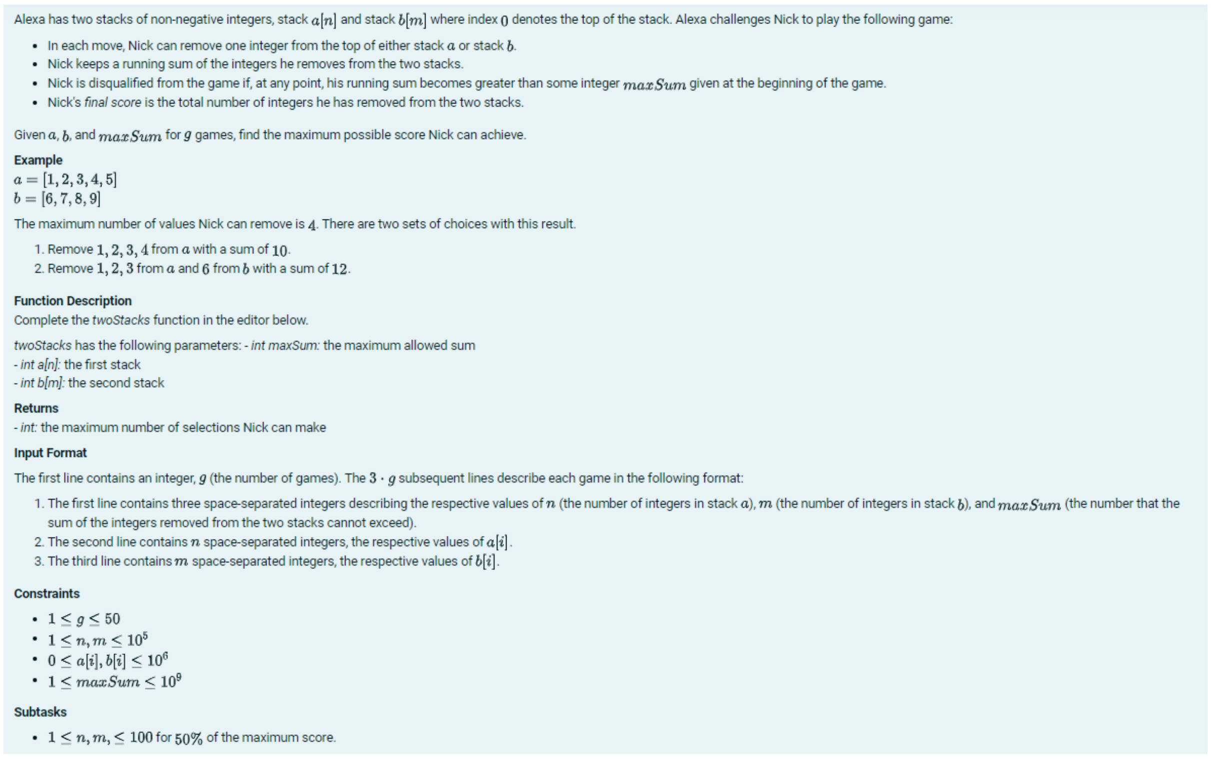
The input of each test case is as follows:

- The first line contains an integer n, the number of elements in the list.
- Each of the next *n* lines contains a data element for a list node.

Constraints

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

2.



Alexa has two stacks of non-negative integers, stack $a[n]$ and stack $b[m]$ where index 0 denotes the top of the stack. Alexa challenges Nick to play the following game:

- In each move, Nick can remove one integer from the top of either stack a or stack b.
- Nick keeps a running sum of the integers he removes from the two stacks.
- Nick is disqualified from the game if, at any point, his running sum becomes greater than some integer $maxSum$ given at the beginning of the game.
- Nick's *final score* is the total number of integers he has removed from the two stacks.

Given a, b, and $maxSum$ for g games, find the maximum possible score Nick can achieve.

Example
$a = [1, 2, 3, 4, 5]$
$b = [6, 7, 8, 9]$

The maximum number of values Nick can remove is 4. There are two sets of choices with this result.

1. Remove $1, 2, 3, 4$ from a with a sum of 10.
2. Remove $1, 2, 3$ from a and 6 from b with a sum of 12.

Function Description
Complete the *twoStacks* function in the editor below.

twoStacks has the following parameters: - *int maxSum:* the maximum allowed sum
- *int a[n]:* the first stack
- *int b[m]:* the second stack

Returns
- *int:* the maximum number of selections Nick can make

Input Format

The first line contains an integer, g (the number of games). The $3 \cdot g$ subsequent lines describe each game in the following format:

1. The first line contains three space-separated integers describing the respective values of n (the number of integers in stack a), m (the number of integers in stack b), and $maxSum$ (the number that the sum of the integers removed from the two stacks cannot exceed).
2. The second line contains n space-separated integers, the respective values of $a[i]$.
3. The third line contains m space-separated integers, the respective values of $b[i]$.

Constraints

- $1 \leq g \leq 50$
- $1 \leq n, m \leq 10^5$
- $0 \leq a[i], b[i] \leq 10^6$
- $1 \leq maxSum \leq 10^9$

Subtasks

- $1 \leq n, m, \leq 100$ for 50% of the maximum score.