Problem A. Merge Linked Lists

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

***Implement Linked list for this question. If found using any other data structure your submission will not be considered.**

Jake managed to get the solution for dsa lab but it is encrypted. To break the encryption he needs to solve this problem.

He is given 2 Linked lists where each Node contains two fields:

- value(Integer)
- Reference to Next Node in linked list

The give linked lists are sorted in ascending order. He has to **merge** both the lists to obtain a **single** list which is also sorted in ascending order.

He is also given a list of M operations which he must perform on the merged list.

There are three operation you can perform on the linked list:

'R': This will reverse the current linked list.

eg if the list is 1 3 4, 'R' will change it to 4 3 1

'I' <pos>(Integer) <val>(Integer): This inserts a new Node with $value = val$ at the index(0-indexed) *pos* in the Linked List.

e.g If the list contains 1 3 4, I 2 5 will insert value 5 at 2nd index(0-indexed) into the linked list and the linked list becomes 1 3 5 4. then I 0 5 will insert at 0th index so new list will be 5 1 3 5 4.

'D' <val>(Integer) <order>(Integer): This will delete the node with $value = val$ from the Linked list. *order* parameter can only be 0 or 1. If *order* is 0, it means the first occurrence of node which satisfies the condition gets deleted . If order is 1 last node with $value = val$ will get deleted.If no node with $value = val$ is found do not do anything.

e.g if the linked list is 1 3 4 3 2 4 6 . D 3 0 will make the linked list 1 4 3 2 4 6, whereas now if D 4 1 is performed, this will make linked list 1 4 3 2 6.

It is guaranteed that pos will be within range of length of linked list at that moment.

Jake is unable to solve the problem. Help him solve it.

Input

First 2 lines contain the linked lists as input stream containing space separated integers ending with -1.

eg 1->2->3 will be given as 1 2 3 -1.

Next line contains an integer M(Number of operations).

Next M line contains instructions that need to be performed over the list.

$0 \le M \le 1000$

$0 \le value \le 10000$

$0 \le pos \le len(linkedlist)$ (At the time of Insertion)

Output

Print space separated node values representing status of linked list after performing M operations onto the linked list. In next line print the sum of all values in the list. If there are no nodes remaining in the linked list Print -1 as output.

Examples

standard input	standard output
1 2 3 4 5 -1 6 7 8 9 -1 4 I 4 3 I 2 5 D 5 1 I 4 3	1 2 5 3 3 4 3 6 7 8 9 51
1 2 3 4 5 -1 6 7 8 9 -1 5 R I 4 3 I 2 5 D 5 1 I 4 3	9 8 5 7 3 6 3 4 3 2 1 51
2 -1 -1 1 D 2 1	-1
1 2 3 -1 4 5 -1 5 I 5 6 I 4 5 D 6 1 I 0 3 D 3 0	1 2 3 4 5 5 20

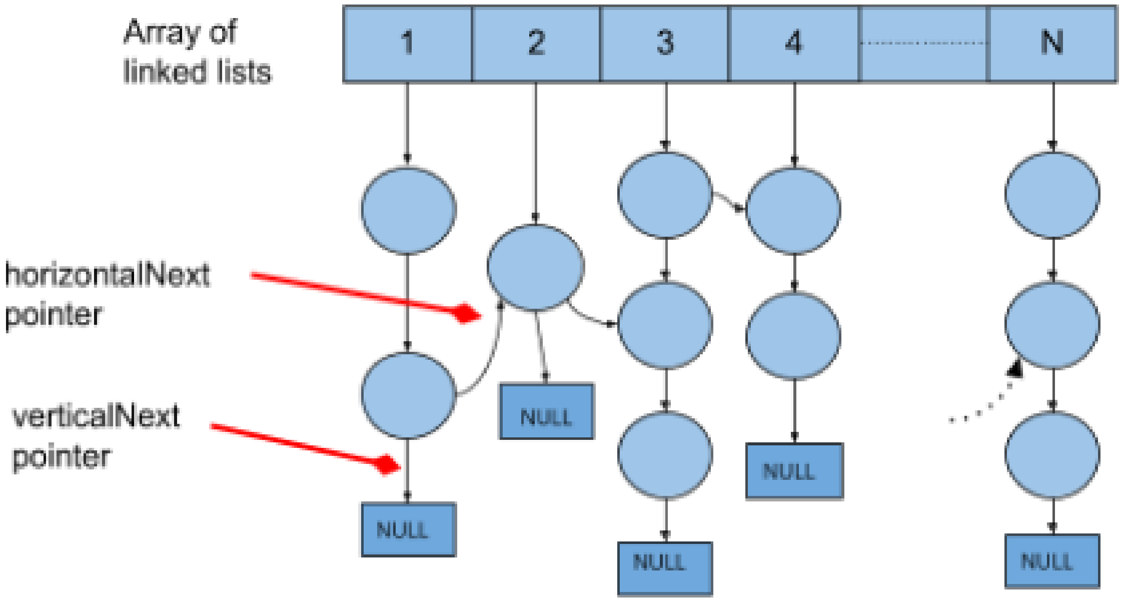
1 3 5 -1
2 4 -1
0
	1 2 3 4 5
	15

Problem B. Overwhelming Lists

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

NOTE: Implement your own linked-lists from scratch. Else your submissions will not be considered.

Lawrence and Hardik were thinking about creating an Overwhelming list, i.e. an array of Linked-Lists. Each element of the array points to the head of a linked list. Each linked list node has 3 attributes namely the *value*, $verticalNext$ pointer, and the $horizontalNext$ pointer. The $verticalNext$ pointer will be pointing to the next node of the same linked list and $horizontalNext$ pointer would point to a node of the next linked list. The Overwhelming list looks like as shown in the following image.



You will be given 3 integers, i, j, p such that $1 \leq i \leq j \leq N$ where N is the length of the Array. Your task is to find if we can reach the node with value p in the j^{th} linked list from any node of the i^{th} linked list. Print "YES"if reachable, else print "NO".

Input

First line contains an integer N, denoting the length of the Array.

In the following N lines, every line's starting integer L denotes the number of elements in that linked list followed by L spaced integers in the same line.

Further you are given $N - 1$ lines to make the connections inside the linked lists.

In the following $N - 1$ lines, every $k^{th}(1 \leq k \leq N - 1)$ line starts with an integer n denoting the number of connections to be made, followed by n pair of integers (a, b) in the same line.

Here (a, b) means the a^{th} node of the k^{th} linked list has the $horizontalNext$ pointer pointing to b^{th} node of the next $(k + 1)^{th}$ linked list.

NOTE: the pair of integers are guaranteed to be given in sorted order.

The last line will contain 3 integers, i, j, p.

Constraints:

- $1 \leq N \leq 1000$
- $0 \leq L \leq 100$
- $0 \leq n \leq L$
- $1 \leq a, b \leq 100$
- $1 \leq i \leq j \leq N$
- $1 \leq p \leq 10^6$
- value of every node is an integer and lies in the range $[1, 10^6]$

Output

Print “YES“ if we can reach the node with value p in the j^{th} linked list from any node of the i^{th} linked list, else print "NO".

Examples

standard input	standard output

3
2 3 6
3 7 8 9
1 10
1 1 2
1 3 1
1 3 10
	YES

4
3 93 74 92
3 36 77 95
9 17 2 50 74 8 91 2 45 16
7 16 46 27 1 98 13 68
0
0
2 5 5 8 1
3 4 98
	YES

4
3 93 74 92
3 36 77 95
9 17 2 50 74 8 91 2 45 16
7 16 46 27 1 98 13 68
0
0
2 5 5 8 6
2 3 98
	NO

4
3 93 74 92
3 36 77 95
9 17 2 50 74 8 91 2 45 16
7 16 46 27 1 98 13 68
0
0
2 5 5 8 6
3 4 68
	YES

4
3 93 74 92
3 36 77 95
9 17 2 50 74 8 91 2 45 16
7 16 46 27 1 98 13 68
0
0
2 2 3 8 6
3 4 16
	NO

1
7 2 50 74 8 91 2 45
1 1 90
	NO

1
7 2 50 74 8 91 2 45
1 1 74
	YES