

932 Checking the N-Queens Problem

The *eight-queens* problem consists of finding (if it exists) a configuration for a set of eight queens on a chessboard, in such a way that no queen is under attack by any other. In other words, there must be a single queen in each row and column of the board, and at most one queen in each diagonal line. The *N-queens* problem is the obvious generalization of this problem to an $N \times N$ board. Consider for instance the following two configurations; the first is a solution to the *5-queens* problem and the second is not:

		Q		
Q				
			Q	
	Q			
				Q

	Q			
Q				
			Q	
	Q			
				Q

Your task is to write a program that, for a given N , determines whether a configuration is a solution to the *N-queens* problem. If not, the program will then check if a solution can be obtained by moving a single queen (in any of the eight possible directions). To simplify, consider that queens can move over each other, i.e, a queen can be moved to any *empty* position in the same row, column, or diagonal line where it stands.

Input

The input consists of several test cases, each of which has:

- a line containing the dimension N of the problem (a positive integer number not greater than 30), followed by
- N lines, each consisting of N characters followed by newline. Characters can only be '0' (zero, corresponding to an empty position) or (capital) 'X', corresponding to a queen. Each line corresponds to a row in the board.
- will contain exactly N occurrences of the character 'X'.

Output

The output for each test case will consist of one of the following:

- a single line containing the word 'YES' (if the configuration is a solution to the *N-queens* problem)
- otherwise, a line containing the word 'NO', followed by a line containing either:
 - the word 'NO', if no solution can be obtained by moving one queen; or
 - the word 'YES' followed by N lines corresponding to the description of the solution discovered, in the same format as in the input.

Print a blank line between test cases.

Sample Input

```
5
00X00
X0000
000X0
0X000
0000X
5
0X000
X0000
000X0
0X000
0000X
```

Sample Output

```
YES

NO
YES
00X00
X0000
000X0
0X000
0000X
```