Problem A. Adjacency

Input file:	`adjacency.in`
Output file:	`adjacency.out`
Time limit:	1 second
Memory limit:	256 megabytes

You are given undirected graph.

Your task is to output its adjacency matrix.

Input

First line contains two integers n and m ($1 \le n \le 100$; $1 \le m \le 10\,000$) — number of vertices and edges in graph, respectively.

Next m lines contain the edges of the graph. Every edge is described by two integer numbers — two vertices connected by this edge.

All vertices are numerated with integers from 1 to n. It's guaranteed that graph doesn't contain parallel edges and loops.

Output

Output n lines. Every line should contain n numbers.

j-th number of i-th line must equal 1, if i and j are connected with an edge, and 0 otherwise.

Examples

adjacency.in	adjacency.out
1 0	0
2 1 1 2	0 1 1 0

Problem B. Tree

Input file:	`tree.in`
Output file:	`tree.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Given undirected graph. Check if this graph is tree.

Input

First line contains two integers n and m — number of vertices and edges in graph, respectively ($1 \le n \le 100$). Next m lines contain edges; i-th of these lines contains two integers u_i and v_i — vertices connected by i-th edge ($1 \le u_i, v_i \le n$).

Graph doesn't contain loops or multiple edges.

Output

Output "`YES`", if given graph is tree, or "`NO`" otherwise.

Examples

tree.in	tree.out
3 2 1 2 1 3	YES
3 3 1 2 2 3 3 1	NO

Problem C. Triangles

Input file:	`triangles.in`
Output file:	`triangles.out`
Time limit:	1 second
Memory limit:	256 megabytes

You are given undirected graph. You have to find number of triangles in it. Triangle is undirected cycle of length 3.

Input

First line contains two integers n and m ($1 \le n \le 100$; $1 \le m \le 10\,000$) — number of vertices and edges in graph, respectively.

Next m lines contain the edges of the graph. Every edge is described by two integer numbers — two vertices connected by this edge.

All vertices are numerated with integers from 1 to n. It's guaranteed that graph doesn't contain parallel edges and loops.

Output

Output one integer — the number of triangles in the graph.

Examples

triangles.in	triangles.out
1 0	0
2 1 1 2	0
3 3 3 2 1 2 3 1	1

Problem D. Distance from the root

Input file:	`rootdist.in`
Output file:	`rootdist.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given rooted tree, and you are to find vertices that are at maximum distance from the root.

Input

First line contains integer n — number of vertices of the tree ($1 \leq n \leq 100$).

Next $n-1$ lines contain parents of vertices in rooted tree. i-th of these numbers is the vertex, that is the root of vertex $i-1$. Vertex 1 is the root of the tree.

Output

Output the maximum distance from root to the vertex one the first line. Distance is the number of edges on the shortest path.

Second line should contain the number of farthest vertices.

And finally the third line should consist of sequence of these vertices in ascending order.

Examples

rootdist.in	rootdist.out
3 1 1	1 2 2 3
3 1 2	2 1 3

Problem E. Looking for cycle

Input file:	`cycle.in`
Output file:	`cycle.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given directed graph.

Your task is to find out, whether the graph contains any cycle. If so, output one of the cycles in the graph.

Input

First line contains two integers n and m ($1 \le n, m \le 100\,000$) — number of vertices and edges in graph, respectively. Next m lines contain the edges of the graph. Every edge is described by two integer numbers — two vertices, edge directed from first of these vertices to the second one. All vertices are numerated with integers from 1 to n.

Output

If there is no cycle in the graph, then output ‘‘`NO`’’.

Otherwise output ‘‘`YES`’’ on the first line, and the sequence of the vertices given in the order of cycle traversal on the second line.

Examples

cycle.in	cycle.out
3 3 1 2 2 3 3 1	YES 2 3 1
3 3 1 2 1 3 2 3	NO

Problem F. Finding path on a grid

Input file:	`dfsongrid.in`
Output file:	`dfsongrid.out`
Time limit:	1 second
Memory limit:	256 megabytes

You are given grid $H \times W$. Some of the cells are blocked. One can move to four of the neighboring cells.

You have to find the path, not necessarily shortest, from cell (x_1, y_1) to cell (x_2, y_2).

Input

First line contains integers W, H, x_1, y_1, x_2, y_2 ($1 \le x_1, x_2 \le W \le 1000$; $1 \le y_1, y_2 \le H \le 1000$).

Each of the next H lines contains W characters. Character "." means that this cell is not blocked, and "*" means blocked.

Cells (x_1, y_1) and (x_2, y_2) are not blocked and don't coincide.

Output

If there is no path between given cells, then output `NO`.

Otherwise, output `YES` on the first line. Then output the sequence of cells, representing the path between given (x_1, y_1) and (x_2, y_2), each cell on separate line.

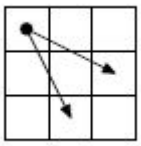
Examples

dfsongrid.in	dfsongrid.out
4 2 1 1 4 2	YES 1 1 1 2 2 2 3 2 4 2
4 2 1 1 4 2 ..*. .*..	NO
4 2 1 1 4 2 ..*. *...	YES 1 1 2 1 2 2 3 2 4 2

Problem G. Knight move

Input file:	`knight.in`
Output file:	`knight.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given rectangular board $n \times m$ (n rows and m columns). There is a chess knight in the top left corner of the board. He needs to be moved to the bottom right corner of the board. In this problem knight can move two cells down and one cell right or one cell down and two cells right.



You are to find, how many different paths are there, from the top left corner to bottom right.

Input

Input contains two integers n and m ($1 \le n, m \le 50$).

Output

Output the number of ways to move the knight from top left to bottom right corner of the board.

Examples

knight.in	knight.out
3 2	1
31 34	293930

Problem H. Tiv tribe

Input file:	`tiv.in`
Output file:	`tiv.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Every year professor Smith goes to Africa to research the tribes that live there. This year he visited tiv tribe. It didn't get much time, so that professor started to understand their language, he learned how to make some ceremonies, but he couldn't understand the numbers written using tiv digits.

Like us, tiv members use positional numeral system with base 10. But digits are different from 0 to 9.

Professor denoted these symbols using letters from 'a' to 'j', but he doesn't know, which letter correspond to each digit. Then the leader of the tribe gave professor the list of the nonnegative numbers, with no leading zeros, written by tiv digits, sorted in strictly ascending order.

Help professor to restore any correct correspondance between letters and digits.

Input

First line contains integer n ($1 \le n \le 10$) — number of numbers in the given list.

Next n lines contain the numbers, each line containing the string consisting of letters 'a' to 'j'. The length of each string doesn't exceed 9.

Output

Output "Yes", if sought correspondance exists. Next line should contain 10 different digits, correspoding to letters 'a' to 'j'. If there are multiple solutions, output any of them.

If professor got something wrong, and answer doesn't exist, output "No" on the first line.

Examples

tiv.in	tiv.out
4 a da dd cc	Yes 7 6 9 8 5 4 3 2 1 0
4 a j jb ac	No

Problem K. Binary Sudoku

Input file:	`bsudoku.in`
Output file:	`bsudoku.out`
Time limit:	1 second
Memory limit:	256 megabytes

Farmer John's cows like to play an interesting variant of the popular game of "Sudoku". Their version involves a 9×9 grid of 3×3 subgrids, just like regular Sudoku. The cows' version, however, uses only binary digits:

```
000 000 000
001 000 100
000 000 000

000 110 000
000 111 000
000 000 000

000 000 000
000 000 000
000 000 000
```

The goal of binary Sudoku is to toggle as few bits as possible so that each of the nine rows, each of the nine columns, and each of the nine 3×3 subgrids has even parity (i.e., contains an even number of `1`s). For the example above, a set of 3 toggles gives a valid solution:

```
000 000 000
001 000 100
001 000 100

000 110 000
000 110 000
000 000 000

000 000 000
000 000 000
000 000 000
```

Given the initial state of a binary Sudoku board, please help the cows determine the minimum number of toggles required to solve it.

Input

- Lines 1 . . . 9: Each line contains a 9-digit binary string corresponding to one row of the initial game board.

Output

- Line 1: The minimum number of toggles required to make every row, column, and subgrid have even parity.

Examples

bsudoku.in	bsudoku.out
000000000 001000100 000000000 000110000 000111000 000000000 000000000 000000000 000000000	3

Note

The Sudoku board in the sample input is the same as in the problem text above.

Three toggles suffice to solve the puzzle.

Problem L. Above the Median

Input file:	`median.in`
Output file:	`median.out`
Time limit:	1 second
Memory limit:	256 megabytes

Farmer John has lined up his N ($1 \leq N \leq 100\,000$) cows in a row to measure their heights; cow i has height H_i ($1 \leq H_i \leq 1\,000\,000\,000$) nanometers (FJ believes in precise measurements)! He wants to take a picture of some contiguous subsequence of the cows to submit to a bovine photography contest at the county fair.

The fair has a very strange rule about all submitted photos: a photograph is only valid to submit if it depicts a group of cows whose median height is at least a certain threshold X ($1 \leq X \leq 1\,000\,000\,000$).

For purposes of this problem, we define the median of an array $A[0 \ldots K]$ to be $A[\lceil \frac{K}{2} \rceil]$ after A is sorted. For example the median of {7, 3, 2, 6} is 6, and the median of {5, 4, 8} is 5.

Please help FJ count the number of different contiguous subsequences of his cows that he could potentially submit to the photography contest.

Input

- Line 1: Two space-separated integers: N and X.
- Lines $2 \ldots N+1$: Line $i+1$ contains the single integer H_i.

Output

- Line 1: The number of subsequences of FJ's cows that have median at least X. Note this may not fit into a 32-bit integer.

Examples

median.in	median.out
4 6 5 10 8 2	8

Note

FJ's four cows have heights 10, 5, 6, 2. We want to know how many contiguous subsequences have median at least 6.

There are 10 possible contiguous subsequences to consider. Of these, only 7 have median at least 6. They are {10}, {6}, {10, 5}, {5, 6}, {6, 2}, {10, 5, 6}, {10, 5, 6, 2}.