Problem E. Black-and-white Square

Input file:	`bwsquare.in`
Output file:	`bwsquare.out`
Time limit:	1 second
Memory limit:	256 megabytes
Detailed Feedback:	**full**

Famous painter Quasimir Malevich painted a picture called "Black-and-white square". The picture looks like a rectangle filled with black and white cells.

Art historian Erik Stripeson put forward a hypothesis that Quasimir painted the picture in the following way: he took a canvas that was initially white, and drew horizontal and vertical black stripes, one cell wide, from the edge to the edge of the canvas.

Association of Computational Art Studies wants to find out whether this hypothesis can be correct. If it can be correct, they also want to know the minimal number of stripes that the painter had to draw in order to paint this picture. Moreover, they need to find a way to draw this picture using this number of stripes.

Input

The first line of the input file contains two integers h and w ($1 \le h, w \le 50$) — the height and the width of the picture.

Each of the next h lines contains w space-separated numbers. Number 0 denotes a white cell, number 1 denotes a black cell.

Output

If Stripeson's hypothesis can't be correct, output only the number `-1`.

If the hypothesis can be correct, the first line of the output file should contain number n — the minimal possible number of stripes needed to draw the picture.

The second line should contain n numbers that describe a way to draw the picture using n stripes. A positive number x stands for drawing a vertical black stripe in x-th column (columns are numbered from left to right, starting with 1). A negative number $-x$ stands for drawing a horizontal black stripe in x-th row (rows are numbered from top to bottom, starting with 1).

If there are several ways to draw the picture using n stripes, output any one of these ways.

Note

Tests with answer `-1` (with numbers from 23 to 27) will be scored only if your solution passes at least one test with number from 12 to 22.

Examples

bwsquare.in	bwsquare.out

`3 4`
`1 0 1 0`
`1 1 1 1`
`1 1 1 1`
	`4`
	`1 3 -2 -3`

`2 2`
`1 0`
`0 0`
	`-1`

Problem F. King's Walk

Input file:	`kingwalk.in`
Output file:	`kingwalk.out`
Time limit:	1 seconds
Memory limit:	256 megabytes
Detailed Feedback:	none

Chess King wants to take a walk in the Spring Field. The Spring Field is a rectangle, and each of its cells contains a single letter of the Latin alphabet.

The King starts his walk in an arbitrary cell of the Field and makes $n - 1$ moves. Each move is made according to the chess rules for the king (to a cell adjacent to the current cell by edge or by vertex). It is not allowed to stay on the same cell as a move. During the walk, the King may visit some cells more than once.

This way, the King will visit n cells. Consider the letters of these cells, in the order they are visited. This is a string s consisting of n letters.

In the evening, the King will compare this string to the motto of his dynasty, which also contains exactly n letters. If in the i-th position, both the motto and the string s contain the same letter, then the King gives promotion to the i-th pawn.

Help the pawns propose a route for the King, so that the number of promoted pawns is as large as possible.

Input

The first line contains two integers h and w ($2 \le h, w \le 20$), the dimensions of the Spring Field.

Each of the h following lines contains w small Latin letters. This is the description of the Spring Field.

The following line contains integer n ($1 \le n \le 50$).

The last line contains the motto of the royal dynasty, a string of n small Latin letters.

Output

The first line of the output should contain m, the maximal possible number of pawns that can be promoted.

The next n lines should contain the description of the optimal route – the coordinates of the cells visited by the King, in the order they are visited.

Coordinates of a cell are the number of its row (rows are numbered from top to bottom, starting with 1), and the number of its column (columns are numbered from left to right, starting with 1).

If there are several optimal routes, output any one of them.

Examples

kingwalk.in	kingwalk.out

3 9
zhautykov
olympiadk
azakhstan
6
almaty
	4
	3 1
	2 2
	1 3
	1 4
	1 5
	1 6

3 9
zhautykov
olympiadk
azakhstan
6
astana
	6
	2 7
	3 6
	3 7
	3 8
	3 9
	3 8

Problem G. High Jump

Input file:	`jumper.in`
Output file:	`jumper.out`
Time limit:	1 second
Memory limit:	256 megabytes
Detailed Feedback:	partial

N jumpers, identified by integers from 1 to N (denoted as ID), participate in the high jump competition. Competition rules are the following.

Jumper has three attempts to clear given height. Attempt is successful if the crossbar remains in place when jumper has left the landing area. Each attempt is done for all jumpers participating in it as follows: jumpers who did not clear the current height yet, of course, if they are not out of the competition, jump in the ascending order of their ID's. Only in the case of three unsuccessful attempts at the current height jumper stops competing. It is forbidden to transfer attempts to the next height, as well as to pass clearing current height or to skip the current attempt.

Winner is the one who clears the greatest height. If two or more jumpers tie for the same place, the tie-breakers are:

1. The fewest misses at the height at which the tie occurred;
2. The fewest misses at the previous height and so on throughout competition;
3. The fewest ID (lower ID corresponds to bigger "rating").

An electronic device recording jumper ID's is installed around the landing area. As a result, the order in which jumpers made jumps is known. Write a program that determines three awarded winners of the competition.

Input

The first line contains N ($3 \le N \le 1000$) — the number of jumpers and M ($9 \le M \le 100\,000$) — the number of all jumps. The second line contains M single-space separated integers — the sequence of jumpers ID's, recorded during the competition.

Output

Output three integers — probable first, second and third place jumpers' ID's, space-separated.

Examples

jumper.in	jumper.out

`5 18`
`1 2 3 4 5 2 3 4 5 2 3 5 1 4 1 4 1 4`
	`1 4 2`

Example Explanation

First and fourth jumpers have cleared starting height at the first round, the others have missed. Winner is the first jumper, as soon as he has cleared his last height first. Third place is given to the second jumper, the one with the minimal ID among the others.

Problem H. Fans

Input file:	`fans.in`
Output file:	`fans.out`
Time limit:	1 second
Memory limit:	256 megabytes
Detailed Feedback:	partial

In the town `M` there are fans of two football teams, `MC` and `MF`. The houses in this town are located along a single street.

The Mayor is tired of fan conflicts, so he has issued an edict about migration of fans of different teams to the different ends of the street. All K fans of `MC` team with their families must move to houses with numbers from 1 to K, and all M fans of `MF` team must move to houses with numbers from $N - M + 1$ to N, where N is the number of houses and families in the town. All other families must move to houses with numbers from $K + 1$ to $N - M$.

The Mayor's assistants have been asked to prepare the migration plan satisfying the following requirements. Each house in the town has its own cost. The compensation will be paid to family which moves to a house with a lower cost, the amount of such compensation is a full cost of the house they originally lived in. If the family moves to a house with a higher or equal cost, there is no compensation for them. No family will move more than once.

Write a program which will determine minimal possible total amount of money required to organize the migration.

Input

The first line of the input file contains an integer N ($2 \le N \le 300$) — the number of houses in the town. N lines follow, each of them containing two integers — the cost of the corresponding house and the number which shows whether there are some fans in the house. The second number can be 0, 1 or 2, where 0 stands for no fans, 1 stands for `MC` fans and 2 — for `MF` fans. It is always guaranteed that there is at least one fan of each team in the town. The costs are positive integers which do not exceed 1000.

Output

Output the minimal possible total amount of money required to organize migration.

Examples

fans.in	fans.out

3
1 2
2 1
3 0
	5

6
5 0
3 1
6 2
5 1
2 1
1 2
	6

Examples Explanation

The first example can be solved as follows: family from 3rd house moves to 2nd, getting compensation of 3 units. 2nd family moves to 1st house, getting compensation 2, and 1st family moves to 3rd house, without any compensation.

The second example can be solved as follows: families from 1st and 4th house exchange their houses without any compensation, and the family from 3rd house exchanges with the family from 5th house with compensation of 6 units.