Back in the day when the famous sixteen-bit three-letter operating system most often used on an 25x80 terminal dominated the PC market, "Nibbles" was everyone's favorite computer game. This problem, however, is not related to Nibbles – it is related to a game called "Connect" which is almost, but not quite, entirely unlike Nibbles.

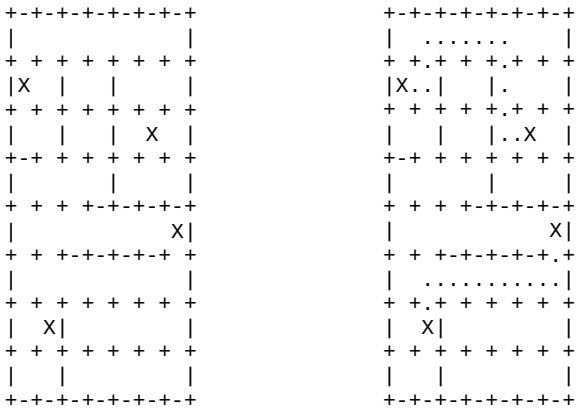
Connect is played on a board composed of squares organized into R rows and C columns, where both R and C are **odd numbers**. Rows and columns are numbered 1 to R and 1 to C, respectively. Each position on the board is either free or blocked by a wall. Additionally, each board satisfies the following constraints:

- Squares with **both coordinates even** are called rooms. They are **never blocked**.
- Squares with **both coordinates odd** are called barriers. They are **always blocked**.
- All other squares are called **corridors**. They may or may not be blocked.
- Corridors **along the edge** of the board are **always blocked**.

Barriers are represented by the '+' (plus) character, blocked horizontal corridors by the '|' (pipe) character, while blocked vertical corridors are represented by the '–' (minus) character. Rooms and free corridors are represented by the blank character.

At the beginning of the game an **even number** of figures (represented by the uppercase letter 'X') are placed on the board – each in a separate **room**. A *path* between figures A and B is a sequence of **free** squares starting from A, ending with B and moving in one of the **four possible directions** in each step (the path includes both endpoint squares, A and B). The length of the path is the total number of steps needed to get from A to B (which is equal to the number of squares contained in the path minus one).

The goal of the player is to first **divide all the figures into pairs**, and then, for each pair, **connect** the two figures with a path. The pairs should be connected in such a way that **no two paths share a common square**. For a completed game, the score is defined as **the sum of the lengths** of all paths.



Write a program that, given the starting position of the Connect game, plays the game in such a way that the **minimum possible score** is achieved.

The test data will guarantee that a solution, although not necessarily unique, will always exist.

INPUT

The first line of input contains two odd integers R and C, ($5 \leq R \leq 25$, $5 \leq C < 80$) – the numbers of rows and columns.

Each of the following R lines of input contains C characters representing one row of the board. Every character will either be one of the three characters '+', '|' and '–', representing a barrier or a blocked corridor, the blank character representing a free corridor or room, or the 'X' character representing a figure in a room.

There will be at least two figures on the board.

Examples of reading an entire line of input (just be sure that the first line with numbers R and C has been read entirely, including newline character):

PASCAL

```
var s : string;
readln(s);
```

C

```
char s[81];
gets(s);
```

C++

```
string s;
getline(cin,s);
```

OUTPUT

The first line of output should contain a single integer, the minimum possible score.

The following R lines of output should contain the description of the board after the game has been played. The board should be formatted exactly the same way as in the input, except that each unoccupied square contained in some path should be represented by the '.' (dot) character.

Note: If there are multiple solutions, you should output any one of them.

GRADING

Partial credit is awarded for incorrect or incomplete solutions that correctly find the minimum possible score.

If the minimum possible score is correct, you will receive 80% of the points for the corresponding test case.

If you choose to find the minimum possible score only, you do not need to output anything else.

EXAMPLES

input

```
17 15
+-+-+-+-+-+-+-+
|               |
+ + + + + + + + +
|X  |   |       |
+ + + + + + + + +
|   |   |  X    |
+-+ + + + + + + +
|       |       |
+ + + +-+-+-+-+-+
|             X|
+ + +-+-+-+-+-+ +
|               |
+ + + + + + + + +
|  X|           |
+ + + + + + + + +
|   |           |
+-+-+-+-+-+-+-+-+
```

output

```
30
+-+-+-+-+-+-+-+
|  .........  |
+ +.+ + + +.+ +
|X..|   |  .  |
+ + + + + +.+ +
|   |   |  X  |
+-+ + + + + + +
|       |     |
+ + + +-+-+-+-+
|             X|
+ + +-+-+-+-+.+
|  ...........|
+ +.+ + + + + +
|  X|         |
+ + + + + + + +
|   |         |
+-+-+-+-+-+-+-+
```

input

```
15 15
+-+-+-+-+-+-+-+
|X|             |
+ + + +-+ + + + +
| |   |X|  X    |
+ + + + + + + +-+
|     | |   |X|
+ +-+-+ + + + +
|       |   | |
+-+-+ + + +-+ +
|     |     | |
+ + + +-+-+ + +
|     |X  | | |
+ +-+-+ + + + +
|    X|       |
+-+-+-+-+-+-+-+
```

output

```
56
+-+-+-+-+-+-+-+
|X|             |
+.+ + +-+ + + +
|.|   |X|  X    |
+.+ + +.+ +.+-+
|.    |.|  .|X|
+.+-+-+.+ +.+.+
|..... .|...|.|
+-+-+.+.+.+-+.+
|    .|...  |.|
+ + +.+-+-+ +.+
|.....|X..| |.|
+.+-+-+ +.+ +.+
|....X|  .....|
+-+-+-+-+-+-+-+
```