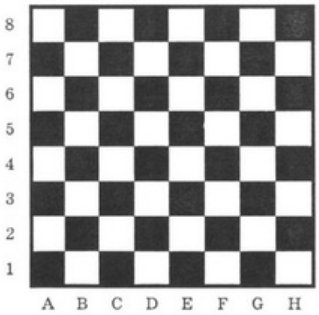
Codeforces Beta Round #3

A. Shortest path of the king

time limit per test: 1 second
memory limit per test: 64 megabytes
input: standard input
output: standard output

The king is left alone on the chessboard. In spite of this loneliness, he doesn't lose heart, because he has business of national importance. For example, he has to pay an official visit to square t. As the king is not in habit of wasting his time, he wants to get from his current position s to square t in the least number of moves. Help him to do this.



In one move the king can get to the square that has a common side or a common vertex with the square the king is currently in (generally there are 8 different squares he can move to).

Input

The first line contains the chessboard coordinates of square s, the second line — of square t.

Chessboard coordinates consist of two characters, the first one is a lowercase Latin letter (from `a` to `h`), the second one is a digit from `1` to `8`.

Output

In the first line print n — minimum number of the king's moves. Then in n lines print the moves themselves. Each move is described with one of the 8: `L`, `R`, `U`, `D`, `LU`, `LD`, `RU` or `RD`.

`L`, `R`, `U`, `D` stand respectively for moves left, right, up and down (according to the picture), and 2-letter combinations stand for diagonal moves. If the answer is not unique, print any of them.

Sample test(s)

input
a8 h1
output
7 RD RD RD RD RD RD RD

B. Lorry

time limit per test: 2 seconds
memory limit per test: 64 megabytes
input: standard input
output: standard output

A group of tourists is going to kayak and catamaran tour. A rented lorry has arrived to the boat depot to take kayaks and catamarans to the point of departure. It's known that all kayaks are of the same size (and each of them occupies the space of 1 cubic metre), and all catamarans are of the same size, but two times bigger than kayaks (and occupy the space of 2 cubic metres).

Each waterborne vehicle has a particular carrying capacity, and it should be noted that waterborne vehicles that look the same can have different carrying capacities. Knowing the truck body volume and the list of waterborne vehicles in the boat depot (for each one its type and carrying capacity are known), find out such set of vehicles that can be taken in the lorry, and that has the maximum total carrying capacity. The truck body volume of the lorry can be used effectively, that is to say you can always put into the lorry a waterborne vehicle that occupies the space not exceeding the free space left in the truck body.

Input

The first line contains a pair of integer numbers n and v ($1 \le n \le 10^5$; $1 \le v \le 10^9$), where n is the number of waterborne vehicles in the boat depot, and v is the truck body volume of the lorry in cubic metres. The following n lines contain the information about the waterborne vehicles, that is a pair of numbers t_i,p_i ($1 \le t_i \le 2$; $1 \le p_i \le 10^4$), where t_i is the vehicle type (1 – a kayak, 2 – a catamaran), and p_i is its carrying capacity. The waterborne vehicles are enumerated in order of their appearance in the input file.

Output

In the first line print the maximum possible carrying capacity of the set. In the second line print a string consisting of the numbers of the vehicles that make the optimal set. If the answer is not unique, print any of them.

Sample test(s)

input
3 2 1 2 2 7 1 3
output
7 2

C. Tic-tac-toe

time limit per test: 1 second
memory limit per test: 64 megabytes
input: standard input
output: standard output

Certainly, everyone is familiar with tic-tac-toe game. The rules are very simple indeed. Two players take turns marking the cells in a 3×3 grid (one player always draws crosses, the other — noughts). The player who succeeds first in placing three of his marks in a horizontal, vertical or diagonal line wins, and the game is finished. The player who draws crosses goes first. If the grid is filled, but neither Xs, nor 0s form the required line, a draw is announced.

You are given a 3×3 grid, each grid cell is empty, or occupied by a cross or a nought. You have to find the player (first or second), whose turn is next, or print one of the verdicts below:

- `illegal` — if the given board layout can't appear during a valid game;
- `the first player won` — if in the given board layout the first player has just won;
- `the second player won` — if in the given board layout the second player has just won;
- `draw` — if the given board layout has just let to a draw.

Input

The input consists of three lines, each of the lines contains characters "`.`", "`X`" or "`0`" (a period, a capital letter X, or a digit zero).

Output

Print one of the six verdicts: `first`, `second`, `illegal`, `the first player won`, `the second player won` or `draw`.

Sample test(s)

input
X0X .0. .X.
output
second

D. Least Cost Bracket Sequence

time limit per test: 1 second
memory limit per test: 64 megabytes
input: standard input
output: standard output

This is yet another problem on regular bracket sequences.

A bracket sequence is called regular, if by inserting "+" and "1" into it we get a correct mathematical expression. For example, sequences "(())()", "()" and "(()(()))" are regular, while ")(", "(()" and "(()))(" are not. You have a pattern of a bracket sequence that consists of characters "(", ")" and "?". You have to replace each character "?" with a bracket so, that you get a regular bracket sequence.

For each character "?" the cost of its replacement with "(" and ")" is given. Among all the possible variants your should choose the cheapest.

Input

The first line contains a non-empty pattern of even length, consisting of characters "(", ")" and "?". Its length doesn't exceed $5 \cdot 10^4$. Then there follow m lines, where m is the number of characters "?" in the pattern. Each line contains two integer numbers a_i and b_i ($1 \le a_i,\ b_i \le 10^6$), where a_i is the cost of replacing the i-th character "?" with an opening bracket, and b_i — with a closing one.

Output

Print the cost of the optimal regular bracket sequence in the first line, and the required sequence in the second.

Print -1, if there is no answer. If the answer is not unique, print any of them.

Sample test(s)

input
(??) 1 2 2 8
output
4 ()()