Problem A. XOR

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

Today Paul and Andrew discovered a new operation, `XOR` of two numbers.

Let us remind you that `XOR`, or *exclusive OR*, is a binary operation which is applied to two integer numbers bitwise. For each bit position, if the bits in the arguments are equal, the resulting bit is `0`, otherwise `1`. For example, 3 `XOR` 5 = 6, because $3_{10} = 011_2$, $5_{10} = 101_2$, so if we apply the operation, the second and the third bits are set to `1`, bit the first bit is set to 0, so we get $110_2 = 6_{10}$.

Paul and Andrew liked this operation so much that they invented a game. First, Paul writes n integer numbers a_i. Second, Andrew writes m integers b_j. After that, Paul finds for each b_j such k that a_k `XOR` b_j is maximal.

The only problem is that Paul is not very fast in finding these numbers. Help him!

Input

The first line of the input contains one integer n ($1 \le n \le 100\,000$) — how many numbers Paul wrote. The second line contains Paul's numbers a_i ($0 \le a_i \le 10^9$). All a_i are different.

The third line contains an integer m ($1 \le m \le 100\,000$) — how many numbers Andrew wrote. The fourth line contains Andrew's numbers b_j ($0 \le b_j \le 10^9$).

Output

Output m numbers: for each b_j, output such a_k that a_k `XOR` b_j is maximal.

Sample input and output

standard input	standard output
2 0 1 2 2 3	1 0
2 3 0 3 2 3 9	0 0 3

Note

Solutions that assume $n, m \le 1000$ will receive at least 40 points.

Problem B. Physical Education

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

There is a physical education lesson in the school. Unfortunately, there are a lot of classes, but only one teacher. So, n classes have the lesson together. Students in every class form a row. In each row girls and boys are mixed.

It is inconvenient to conduct a lesson for the set of rows, so the teacher decided to combine all rows in one. He can take two rows and place all students from one of them at the end of another, so it will be one long row. It is not allowed to change the order of students inside rows.

After the teacher forms one single row, the students are divided into two groups as follows. The students say in the order they appear in the row : "First", "Second", "First", "Second", etc. So the students who stand on odd places form the "First" group, and students who stand on even places form the "Second" group. The "First" group go to play football and a "Second" group go to play volleyball.

Boys like to play football, and girls don't. Therefore, the teacher wants to arrange students in such a way so the number of boys playing football will be as large as possible. Help him find the maximum number of boys that can go play football.

Input

The first line of input contains one integer number n ($1 \le n \le 100\,000$) — the number of classes.

The next n lines contain descripions of initial rows. Each line is a string of letters `B` and `G`. The letter `B` means a boy, and the letter `G` means a girl. The first letter in a string corresponds to the first student in a row. Each row is not empty and contains at most 100 letters.

The total number of students does not exceed 10^6.

Output

Output single integer number — the maximum possible number of boys playing football.

Sample input and output

standard input	standard output
4 GBG BB GG BGG	3

Note

In the sample above, it is not possible for all boys to play football in the same time, but if we arrange the rows as follows: `BGG`, `BB`, `GG`, `GBG`, then three of four boys will play football.

For testcases, which are together worth at least 60 points, $n \le 1000$.

For testcases, which are together worth at least 30 points, $n \le 10$.

Problem C. Competition

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

The programmers of Xednay are constantly developing new software products. One of the stages of testing a new program is verification of the correct conclusion in different conditions. And one of the conditions is the correct exit of all procedures and functions that ran during the program. To check if everything was right and the program didn't stop working in the middle of some routines, programmers created a tool which is logging all program activity.

The output this utility produce is very simple — it is a string of round, square and curly brackets. The opening bracket means the launch of a new procedure with certain properties, and the closing bracket means the completion of a procedure with the same properties that the corresponding opening one.

Clearly, looking at such string, it is easy to see whether there were any problems. You need only to verify the bracket sequence to be correct.

The competitors of Xednay, the Elgoog, is not standing still. They create a virus which changes the result of the above utility by changing exactly one symbol in it.

When it was discovered, you was commissioned to write a program, which restores the string. The program should find a substitution of one character in the bracket sequence, after which it become correct.

Input

The first and only line of the input contains the bracket sequence. It is not longer than 10^5 symbols. The only allowed symbols are `(`, `)`, `[`, `]`, `{` and `}`.

Output

If it is possible to change exactly one symbol in the given bracket sequence to make it correct, output the restored sequence. If it is not, output the string "No solution". If there are several such sequences, output any one.

Sample input and output

standard input	standard output
`(){[()())}`	`(){[()()]}`
`[[[]`	`[[]]`
`()))(`	`No solution`
`([])`	`No solution`

Note

For testcases, which are together worth at least 60 points, the length of the sequence is not more than 1000 symbols.

Problem D. Parties

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

In Byteland, there are n cities connected by highways. Each highway is bidirectional and has a certain length. There are also two parties. The first one declares that, when having dinner, everybody must keep a fork in the left hand, the second one is strictly for the right hand. Each of the cities in Byteland have chosen exactly one of these parties to support. However, from time to time, revolutions take place in this or that city, and the other party takes place of its opponent.

The government of Byteland has offered you to write a program which, for every moment of time, should estimate the political stability in the country. The mathematical model, developed by the leading scientists, shows that the stability depends on how close are the cities which support the same party, because the closer they are, the more is the probability that they will form a coalition.

So the program that you are to write must, given the information about all the cities, highways and revolutions, find, after each revolution, two cities that support the same party and having the smallest distance between them.

Input

The first line of the input contains three integers n, m и k ($1 \le n, m, k \le 100\,000$) — the numbers of cities, highways and revolutions, correspondingly.

The second line contains a string s of length n, which describes the political situation at the initial moment. At i-th position, the symbol `L` denotes that i-th city supports the left-hand-party, and the symbol `R` is for the right-hand one.

The next m lines contain three integers each: a_i, b_i и l_i ($1 \le a_i, b_i \le n$, $a_i \ne b_i$, $1 \le l_i \le 10^9$) — the numbers of the cities connected by the i-th highway, and the length of this highway. Each pair of the cities is connected by not more than one highway.

The next k lines contain the number of cities c_j ($1 \le c_j \le n$), where revolutions happened, in the order of these events. Because the cities are, in general, quite conservative, in every city there have been no more than one revolution, so all c_j are different.

Output

Output $k + 1$ reports about the closest cities that may form a coalition. The first report must correspond to the initial moment, and every next report must describe the situation after next revolution.

Every report must be output on a separate line and must contain three integers d_i, x_i и y_i — the minimal distance between the cities that support the same party, and the numbers of these cities, respectively. If there are several such pairs of cities, output the data for any of them. It is guaranteed that there will be at least one such pair.

Sample input and output

standard input	standard output
5 6 4 LRLRL 1 4 1 2 3 2 3 4 3 4 5 4 2 5 5 2 4 6 1 4 3 5	4 1 3 1 4 1 3 3 4 2 3 2 2 2 3

Note

Solutions assuming that $n \le 100$ will receive at least 30 points.

Solutions assuming that $n \le 2000$ will receive at least 60 points.