

## Problem A. Alice and Path

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

Consider an infinite planar board consisting of equal equilateral triangles. Alice was standing in such a triangle, facing one of its vertices. After that, she made a sequence of steps. There are three kinds of steps. Step “l” is to turn left and move to the neighboring triangle. Step “r” is to turn right and move to the neighboring triangle. Step “b” is to turn around and move to the neighboring triangle. Note that, after every possible step, Alice again faces one of the vertices of her new triangle: the girl looks at the vertex opposite to the side she just stepped over.

So, Alice went along some path consisting of steps “l”, “r”, and “b”. Help her return to the triangle where she started! Compose a sequence of steps which will bring her there. The direction in which Alice will look after completing the steps is irrelevant: the important thing is to finish at the initial triangle.

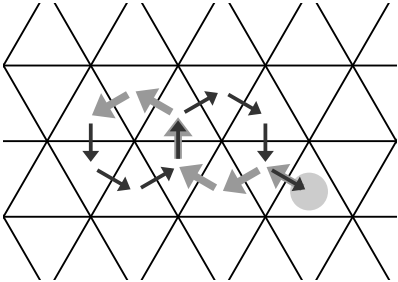
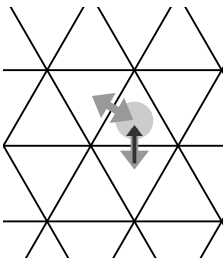
### Input

The input consists of a single line denoting the sequence of steps: it can contain the letters “l”, “r”, and “b”, and has length from 1 to 10 000 letters. It is guaranteed that completing this sequence of steps brings Alice to a triangle different from the initial one.

### Output

Print a line containing a sequence of steps. It must have length from 1 to 100 000 letters and consist of the letters “l”, “r”, and “b”. After completing the input sequence of steps, and then the output sequence, Alice must finish at the initial triangle. If there are several possible answers, print any one of them. Note that the length of the output sequence need not be the minimum possible.

### Examples

standard input	standard output	Notes
llrrll	llllrrrl	
lbr	b	

## Problem B. John and the Magic Box

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

*This is an interactive problem.*

John has an array of  $n$  mysterious integers. He has an access to a magic box that can combine two integers into one. Let  $x \circ y$  be the result of combining two integers,  $x$  and  $y$ , with this magic box. After a lot of experiments, John has noticed that the magic box has the following properties:

- $x \circ y = y \circ x$
- $x \circ (y \circ z) = (x \circ y) \circ z$

John has  $q$  cousins, and each of them likes all the integers in his array except some  $k$  of them. John wants to give gifts to all his cousins, so he wants to give each cousin the combination of all his integers except the  $k$  this cousin doesn't like.

John likes his cousins, but his magic box is old and worn off because of his intense experiments. He is willing to use the box at most  $4(n + q + q \cdot k)$  times. Help him get all the required combinations!

### Interaction Protocol

Initially, you are given a line containing three integers  $n$ ,  $q$  and  $k$  ( $1 \leq q \leq n \leq 2000$ ,  $n \geq 3$ ,  $2 \leq 2^k \leq n$ ). Next, you are given a line containing  $n$  integers  $a_1, \dots, a_n$  ( $0 \leq a_i < 2^{50}$ ). After that, you can use three types of queries (each query must be written on a separate line):

- “?  $x$   $y$ ” where  $x$  and  $y$  are integers ( $0 \leq x, y < 2^{50}$ ). A query of this type asks to use the magic box. The response to this query is a line containing the integer  $x \circ y$  ( $0 \leq x \circ y < 2^{50}$ ). You can only use a query of this type  $4(n + q + q \cdot k)$  times in total. If you exceed this amount, the jury program will print “-1”, and after that, the checking will be terminated.
- “next”. A query of this type asks for the indices of  $k$  integers which the next cousin does not like. The response to this query is a line containing  $k$  distinct integers  $d_1, \dots, d_k$  ( $1 \leq d_i \leq n$ ): the indices of elements which this cousin does not like. *You can only use a query of this type again after you answer the previous such query.*
- “!  $x$ ” where  $x$  is an integer ( $0 \leq x < 2^{50}$ ). Here,  $x$  should be the integer John will give to the last described cousin (that is, the cousin corresponding to the most recent “next” query). There is no response for a query of this type.

To prevent output buffering, flush the output buffer after each printed line: this can be done by using, for example, `fflush (stdout)` in C or C++, `System.out.flush ()` in Java, `flush (output)` in Pascal, or `sys.stdout.flush ()` in Python. Also, do not forget to terminate each line of output with a newline character.

## Example

standard input	standard output
3 3 1 0 1 0  1  1  2  0  3	   next  ? 1 0  ! 1 next  ? 0 0  ! 0 next  ! 1

## Note

In each test, the rules for the magic box are fixed and don't depend on your queries. Different rules are used for different tests. It is guaranteed that the magic box satisfies the conditions from the problem statement.

In the sample test, the operation performed by the magic box is assumed to be bitwise OR.

## Problem C. Anti-Distance

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

Consider a plane partitioned into squares with side 1. Let us choose a square and draw coordinate axes from its center parallel to its sides.

Each fifth square contains an obstacle: more precisely, the obstacles are placed in all squares with centers at points  $(2i + j, i - 2j)$  for all possible integer  $i$  and  $j$ . The exact placement of obstacles is visualized in the example notes below. All other squares are free.

Amelia stands in some free square  $A$  and wants to move to some free square  $B$ . In one step, she can move from a square to one of its neighbors: the squares sharing a side with it, but only if the corresponding neighboring square is free. What is the minimum possible number of steps Amelia must make to arrive to square  $B$ ?

### Input

The first line contains two integers  $x_1$  and  $y_1$ , the coordinates of the initial square  $A$ . The second line contains two integers  $x_2$  and  $y_2$ , the coordinates of the destination square  $B$ . All given coordinates are between 1 and  $10^9$ . It is guaranteed that both given squares are free.

### Output

Print one integer: the minimum possible number of steps from the initial square to the destination square.

### Examples

standard input	standard output	Notes
1 1 5 2	7	
1 1 2 5	5	

## Problem D. Machines on the Moon

Input file: *standard input*  
Output file: *standard output*  
Time limit: 12 seconds  
Memory limit: 256 mebibytes

Jeff is an inventor. Throughout his life, he invented various machines. He never gets patents for them because he has never trusted the government.

A machine of order  $\ell$  is an acyclic directed graph with  $\ell$  sources (vertices with no incoming edges) and one selected sink (vertex with no outgoing edges). Each vertex of the graph except sources has exactly two inner edges. One inner edge is referred to as the left edge, and another one is referred to as the right edge.

Each vertex  $v$  of the graph computes a function  $f_v : \{0, 1\}^\ell \rightarrow \{0, 1\}$ . In particular,  $i$ -th source  $s_i$  computes the function  $f_{s_i}(x_1, \dots, x_n) = x_i$ . Each inner vertex  $v$  is labeled with the function  $b_v : \{0, 1\}^2 \rightarrow \{0, 1\}$ . Let  $u$  be the starting vertex of the left inner edge to  $v$ , and  $w$  be the starting vertex of the right inner edge. The function  $f_v$  is then defined as

$$f_v(x) = b(f_u(x), f_w(x)).$$

Jeff says that the function computed by a machine is the function computed by its selected sink.

One problem keeps Jeff awake at night for a couple of months. Jeff has an undirected graph  $G = (V, E)$  which his uncle John left him. Jeff is really puzzled with cliques and independent sets in this graph.

A set  $S \subseteq V$  is a *clique* of the graph  $G = (V, E)$  if, for every pair of distinct vertices  $u, v \in S$ , we have  $(u, v) \in E$ .

A set  $S \subseteq V$  is an *independent set* of the graph  $G = (V, E)$  if, for every pair of distinct vertices  $u, v \in S$ , we have  $(u, v) \notin E$ .

Jeff has noticed that if  $I$  is an independent set and  $C$  is a clique, then  $|I \cap C| \leq 1$ . But it still leaves two options: either  $I \cap C = \emptyset$  or  $|I \cap C| = 1$ .

Let  $n = |V|$ . Jeff denotes a vector corresponding to the set  $S$  as  $\chi_S \in \{0, 1\}^n$ : the  $i$ -th component of  $\chi_S$  is 1 if  $i \in S$  and 0 otherwise.

Jeff wants to build two machines of order  $n + 2k$ , where  $n = |V|$  and  $k$  is Jeff's favorite positive integer, such that they would be able to decide if  $I \cap C = \emptyset$  **together**.

The first machine receives a vector  $\chi_C \in \{0, 1\}^n$ , where  $C$  is a clique, followed by  $2k$  zeroes. The second machine receives a vector  $\chi_I \in \{0, 1\}^n$ , where  $I$  is an independent set, followed by  $2k$  zeroes. After that, the machines will work together using the scheme described below and decide if  $I \cap C = \emptyset$ .

Let us denote the first machine as  $A$  and identify it with the function it computes. For example, we denote the result of the first execution as  $A(\chi_C, \underbrace{0, \dots, 0}_{2k}) \in \{0, 1\}$ . Similarly, we denote the second machine as  $B$ .

Jeff has developed a scheme to make the machines work together. Let

$$a_0 = A(\chi_C, \underbrace{0, \dots, 0}_{2k}),$$

$$b_0 = B(\chi_I, \underbrace{0, \dots, 0}_{2k}).$$

Then let

$$a_i = A(\chi_C, a_0, b_0, a_1, b_1, \dots, a_{i-1}, b_{i-1}, 1, \underbrace{0, \dots, 0}_{2(k-i-1)-1}),$$

$$b_i = B(\chi_I, a_0, b_0, a_1, b_1, \dots, a_{i-1}, b_{i-1}, 1, \underbrace{0, \dots, 0}_{2(k-i-1)-1}).$$

If  $(a_{k-1} = 0) \vee (b_{k-1} = 0)$ , Jeff considers the answer of the machines to be  $I \cap C = \emptyset$ , and if  $(a_{k-1} = 1) \wedge (b_{k-1} = 1)$ , Jeff considers the answer of the machines to be  $I \cap C \neq \emptyset$ .

Jeff struggles to develop such machines. Help him!

## Input

The first line contains three integers  $n$ ,  $m$ , and  $k$ : the number of vertices in  $G$ , the number of edges in  $G$ , and Jeff's favorite positive integer ( $1 \leq n \leq 1000$ ,  $1 \leq m \leq 10\,000$ ,  $k \geq (\lceil \log_2(n) \rceil + 1)^2$ ,  $k \leq n$ ). The next  $m$  lines contain the description of the edges of  $G$ . Each of these lines contains two integers  $a$  and  $b$  ( $1 \leq a, b \leq n$ ,  $a \neq b$ ).  $G$  does not contain parallel edges.

## Output

Print the descriptions of both machines.

A description of a machine should have the following format. The first line of the description should contain one integer  $t$ : the number of nodes in the machine ( $n + 2k + 1 \leq t \leq 1\,500\,000$ ). The nodes are numbered from 0 to  $t - 1$ . You should print the descriptions of the non-source nodes. The  $i$ -th of the next  $t - (n + 2k)$  lines should contain the description of the node numbered  $(i + (n + 2k) - 1)$ . The line should contain two integers  $x$  and  $y$ : the left and the right input nodes for node  $i + (n + 2k) - 1$  ( $0 \leq x, y < i$ ), followed by a string  $f = f_0 f_1 f_2 f_3$ . The string  $f$  describes the function  $b$ :  $f_0 = b(0, 0)$ ,  $f_1 = b(0, 1)$ ,  $f_2 = b(1, 0)$ ,  $f_3 = b(1, 1)$ . The vertex  $t - 1$  is the selected sink of the machine.

**This problem is technically interactive (it is sadly so hard to find a clique or an independent set in a graph!), so you need to flush the output after printing the answer to prevent buffering.**

## Example

standard input	standard output
2 1 5 1 2	20 2 2 1100 3 3 1100 2 3 0001 1 14 0001 2 13 0001 16 0 0001 12 15 0111 18 17 0111 16 2 2 1100 12 1 0001 2 4 0001 13 14 0111

## Note

The answer is checked in the following way:

- The jury picks a clique  $C$  and an independent set  $I$  from the given graph.
- The jury computes the values  $a_0, b_0, a_1, b_1, \dots, a_{k-1}, b_{k-1}$  in that order and checks if the boolean expressions  $(a_{k-1} = 0) \vee (b_{k-1} = 0)$  and  $I \cap C = \emptyset$  have the same value.
- The process is repeated for several pairs  $(C, I)$ .

## Problem E. Coin Tournament

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

There is a coin tossing tournament organized by the Thieves Guild. A total of  $x$  thieves and  $y$  assassins are going to take part in the tournament. Initially, each participant has a position denoted by an integer from 1 to  $x + y$ . The games happen while there are at least two participants. In each game, consider participant  $A$  standing at the position with the greatest number. Let it be position  $k$ . Participant  $A$  tosses a fair coin, hoping to move to position  $\lfloor k/2 \rfloor$  which is occupied by some participant  $B$  at the moment. If  $A$  got heads, then  $A$  moves to  $B$ 's position, and  $B$  is kicked out of the tournament. If  $A$  got tails, then  $A$  is kicked out of the tournament, and  $B$  remains at the same position. The last remaining participant is the winner.

Делегация ассасинов опоздала к регистрации, так что воры заняли позиции от 1 до  $x$ , и ассасинам остались позиции от  $x+1$  до  $x+y$ . Казначей турнира хочет заранее знать, какова вероятность победы ассасина на турнире, если во всех играх используется идеальная монетка, то есть вероятности выпадения «орла» и «решки» равны  $1/2$  и не зависят друг от друга. Найдите эту вероятность.

The delegation of assassins was late for the registration, so the thieves already occupied the positions from 1 to  $x$ , and the assassins were left with the positions from  $x + 1$  to  $x + y$ . The tournament treasurer wants to know in advance what is the probability of an assassin winning the tournament, given that a fair coin is used for every game, that is, the probabilities of heads and tails are equal to  $1/2$ , and all coin tosses are independent. Find this probability.

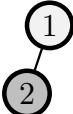
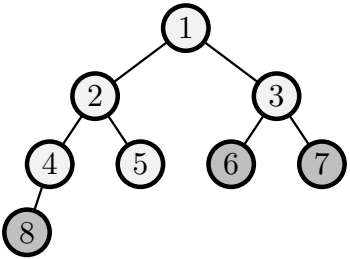
### Input

The first line contains two integers  $x$  and  $y$ : the number of thieves and the number of assassins ( $1 \leq x, y \leq 1\,000\,000$ ).

### Output

Output the required probability as a decimal fraction. Your answer will be considered correct if the absolute or relative error will be less than  $10^{-6}$ .

### Examples

standard input	standard output	Notes
1 1	0.5	
5 3	0.312500000000	

## Problem F. Fizz and Buzz

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

Two dwarves, Fizz and Buzz, compose a sequence of  $n$  integers. Each element is constructed as follows. First, the friends toss a fair coin to choose who will create the next number: with probability  $\frac{1}{2}$  it will be Fizz, and with probability  $\frac{1}{2}$  it will be Buzz. After that, if the toss was in favor of Fizz, he considers all integers from 1 to 10 000 divisible by 3 and selects one of them at random. All these integers have the same probability of being selected. If the toss was in favor of Buzz, he considers all integers from 1 to 10 000 divisible by 5 and selects one of them at random. All these integers also have the same probability of being selected. All coin tosses and all integer selections are independent events.

The dwarves composed a sequence of  $n$  integers by the rules outlined above. Given the resulting sequence, guess who created which number, and do not make too many errors in the process.

### Input

The first line contains an integer  $n$ , the length of the sequence ( $1 \leq n \leq 10\,000$ ). The second line contains the sequence itself: the integers  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10\,000$ ). It is guaranteed that the sequence was composed according to the problem statement using a random number generator.

### Output

Print a single line with  $n$  characters each of which is either “F” or “B”. In the perfect answer, the character at position  $i$  must be “F” if the number  $a_i$  was created by Fizz, or “B” if the number  $a_i$  was created by Buzz. Your answer can contain at most 1200 errors. In other words, it can differ from the perfect answer at no more than 1200 positions.

### Example

standard input	standard output
10 3216 7770 8448 3438 6255 330 405 5835 6140 7475	FBFFBBFBBB

### Explanation

The example above shows the perfect answer. In this example, the solution can not make more than ten errors, so any correctly formatted answer will be accepted.



## Problem G. Game on Tree

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

You are given an undirected rooted tree. Vertices are numbered with integers from 1 to  $n$ . The root is vertex 1.

Two players are playing a game on this tree. They make alternating moves.

The first player can mark one leaf on his move, and it will remain marked until the end of the game. A leaf of a rooted tree is a non-root vertex with only one neighbor. Initially, all leaves are unmarked.

The second player controls a chip. The chip is always located in some vertex. Initially, the chip is placed in vertex 1, the root of the tree. On his move, the second player can put the chip in any vertex adjacent to the current one, or leave it in the current vertex.

The game ends when either all leaves are marked (the first player wins) or the chip is put into some unmarked leaf (the second player wins). Who will be the winner if both players play optimally?

### Input

The first line contains an integer  $n$ : the number of vertices in the tree ( $2 \leq n \leq 100\,000$ ). The second line contains  $n - 1$  integers  $p_2, p_3, \dots, p_n$ . Here,  $p_i$  is the parent of vertex  $i$  ( $1 \leq p_i < i$ ).

### Output

If the first player wins, print “FIRST” on the first line. After that, on the second line, print an integer  $v$  ( $2 \leq v \leq n$ ): the number of vertex the first player has to mark on the first move. This vertex must be a leaf. The first player must win after this move if both play optimally. In case there are several such vertices, print any one of them.

If the second player wins, print “SECOND” on the first line.

### Examples

standard input	standard output
2 1	FIRST 2
3 1 1	SECOND

## Problem H. Jack and Jill

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

*This is an interactive problem.*

Jack and Jill play “guess the number”. Initially, Jill has to choose a secret number: an integer from 1 to  $10^9$ . After that, Jack asks questions in the form “is it the number  $x$ ?” with some integer  $x$  from 1 to  $10^9$ . For each question, Jill has to answer either “yes”, or “no, my number is greater”, or “no, my number is less”. The game ends when Jack correctly guesses the secret number, or after 100 questions if it does not happen by then.

Despite Jack’s best efforts, he was not able to guess the number in less than 30 questions. He realized that Jill is cheating: instead of choosing a secret number in advance, she answers the questions in such a way that the game lasts long enough. Jack pondered: how does she do that?

This is an interactive problem: you play as Jill, and the jury plays as Jack. Your task is to answer the questions in such a way that Jack asks at least 30 questions before the game ends. Keep in mind that your answers should not contradict each other: otherwise, Jack will immediately call you out on it!

### Interaction Protocol

The game consists of steps. Each step starts with the jury giving an integer  $x$  ( $1 \leq x \leq 10^9$ ) on a separate line: the number for which Jack asks “is  $x$  the secret number?”. In response, your solution must print one character on a separate line: “=” if Jill’s reply is “yes”, or “>” if Jill’s reply is “no, my number is greater”, or “<” if Jill’s reply is “no, my number is less”.

After printing the line with the character, flush the output buffer: this can be done by calling, for example, `fflush (stdout)` in C or C++, `System.out.flush ()` in Java, `flush (output)` in Pascal, or `sys.stdout.flush ()` in Python.

If Jill’s answers contradict each other, or Jill answers “yes”, or 100 steps were made, the game finishes immediately, and your solution must terminate gracefully. Your solution passes a test if the answers were not contradictory and the number of steps was at least 30.

Jack uses different playing strategies in different tests.

### Examples

standard input	standard output
1	>
2	>
...and so on...	...and so on...
29	>
30	=
1000	<
999	<
...and so on...	...and so on...
902	<
901	<

### Explanations

In the first example, Jack says numbers 1, 2, 3, and so on. In the second example, Jack says numbers 1000, 999, 998, and so on. He is guaranteed to follow these two strategies in the first two tests.

In both examples, Jill just chose 30 as the secret number in advance. Your solution may of course use any other strategy.

## Problem I. Laws

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

There are two inhabitants in the Faraway Kingdom: Alyonushka the Peasant and Ivanushka the King. Alyonushka works on a farm, and Ivanushka makes laws.

Alyonushka has  $x$  coins. Each day, Alyonushka gets one more coin from the treasury for her work. The amount of coins in the treasury can be considered infinite.

If the number of Alyonushka's coins divides evenly by two, Ivanushka can make another peasant law, and Alyonushka will be allowed to keep only half of her coins: the other half immediately goes to the treasury. If the number of Alyonushka's coins divides evenly by three, Ivanushka can make another farm law, and Alyonushka will be allowed to keep only one third of her coins: the other two thirds immediately go to the treasury. Ivanushka can make new laws at any moment, in any order, and do it any number of times.

Today Ivanushka got angry with Alyonushka. Now he wishes Alyonushka to have only one coin left. What is the minimum possible number of days required to achieve that?

### Input

The first line of input contains an integer  $x$ : the initial number of Alyonushka's coins ( $1 \leq x \leq 10^9$ ).

### Output

On the first line, print  $t$ : the minimum possible number of days required for Ivanushka to leave Alyonushka with only one coin. On the second line, print a sequence of integers: any possible sequence of events which takes  $t$  days and transforms  $x$  coins into 1. The sequence must start with  $x$  and end with 1. Every two consecutive numbers  $u$  and  $v$  in the sequence must satisfy either  $v = u + 1$  (a day has passed),  $v = u/2$  (a new peasant law has been made), or  $v = u/3$  (a new farm law has been made).

### Examples

standard input	standard output
11	1 11 12 6 2 1
100	2 100 50 25 26 27 9 3 1
1	0 1

## Problem J. Cubic Path

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

A *path* in the  $n$ -dimensional set  $\{0, 1\}^n$  is a sequence of  $n$ -dimensional points  $x_1, x_2, \dots, x_k \in \{0, 1\}^n$  such that, for each  $i$  ( $1 \leq i \leq k - 1$ ), points  $x_i$  and  $x_{i+1}$  differ in exactly one coordinate, and all the points  $x_1, \dots, x_k$  are distinct. The length of the path  $x_1, \dots, x_k$  is  $k$ .

A path  $x_1, \dots, x_k$  is *imperfect* if there exists a shorter path  $y_1, \dots, y_\ell$  which leads from the first to the last point of this path and consists of a subset of the same points. In other words,  $\{y_1, \dots, y_\ell\} \subseteq \{x_1, \dots, x_k\}$ ,  $x_1 = y_1$ ,  $x_k = y_\ell$  and  $\ell < k$ . If a path is not imperfect, it is *perfect*.

Your task is to find the longest perfect path in the set  $\{0, 1\}^n$ .

### Input

The only line contains a single integer  $n$  ( $1 \leq n \leq 6$ ).

### Output

On the first line, print  $L$ , the length of the path. On the next  $L$  lines, print the description of the path  $x_1, \dots, x_L$ :  $i$ -th of these lines must contain  $n$  characters (zeroes and ones) describing the point  $x_i$ .

It is easy to see that there are multiple longest perfect paths. Print any one of them.

### Examples

standard input	standard output
2	3 00 01 11
3	5 000 001 011 111 110
4	8 0000 0001 0011 0111 0110 1110 1100 1101

## Problem K. Red-Black Tree

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 256 mebibytes

Consider a binary rooted tree: a vertex is selected as the root, all edges go in the direction from the root, and each vertex has zero, one, or two children. We will say that each vertex has exactly two outgoing edges: in case it has less than two children, the remaining edges go into the void.

We will say a tree is *red-black* if the following conditions are satisfied:

- Each vertex is colored either red or black.
- There is no edge in the tree which has two red endpoints. The void is considered black.
- Consider all paths along the edges which go from the root to the void. The number of black vertices is the same on all such paths.

A similar coloring scheme can be used in binary search trees to make them balanced.

You are given a non-empty binary rooted tree. Color its vertices so that it becomes a red-black tree, or determine that it is impossible.

### Input

The first line contains an integer  $n$  ( $1 \leq n \leq 500$ ), the number of vertices in the tree. The vertices are numbered by integers from 1 to  $n$ .

The next line contains  $n$  space-separated integers:  $p_1, p_2, \dots, p_n$  ( $0 \leq p_i \leq n$ ). A number  $p_i > 0$  means that vertex  $i$  is a child of vertex  $p_i$ . In case  $p_i = 0$  means that  $i$  is the root of the tree.

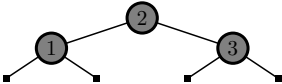
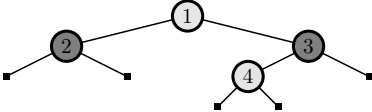
It is guaranteed that the input specifies a valid binary rooted tree: there is exactly one root, each vertex has from 0 to 2 children, and it is possible to arrive at any vertex by starting at the root and moving along the edges.

### Output

If it is possible to color the tree so that it becomes a red-black tree, print any such coloring as a line of  $n$  characters. The character at  $i$ -th position must be "R" if vertex  $i$  is red, or "B" if it is black.

If it is not possible to color the tree, print "Impossible".

### Examples

standard input	standard output	Notes
3 2 0 2	BBB	
4 0 1 1 3	RBRR	
4 4 1 1 0	Impossible	