## Problem A. AntiGray Codes

| | |
|---|---|
| Input file: | `antigray.in` |
| Output file: | `antigray.out` |
| Time limit: | 2 second |
| Memory limit: | 64 megabytes |

Peter Q. Perverse is tired of Gray codes. So he has developed his own codes, called AntiGray codes. The AntiGray codes consist of all ternary numbers of length $n$. The only conditions are that each number differs from previous in all positions and every number occurs exactly once. Your task is to say whether the AntiGray codes for a given $n$ exist, and if the answer is positive, to sample example AntiGray codes.

### Input

The input file consists of one integer number $n$ ($1 \leqslant n \leqslant 11$).

### Output

If AntiGray codes for a given $n$ do not exist, output a single word `IMPOSSIBLE`. Otherwise output all ternary numbers of length $n$ in AntiGray order. Output file must not contain any spaces, each ternary number must consist of exactly $n$ digits.

### Example

| antigray.in | antigray.out |
|---|---|
| 1 | 1<br>0<br>2 |

## Problem B. Count Primes

| | |
|---|---|
| Input file: | `countpr.in` |
| Output file: | `countpr.out` |
| Time limit: | 10 seconds |
| Memory limit: | 256 mebibytes |

Count the number of primes between $A$ and $B$, inclusive. $1 \leqslant A \leqslant B \leqslant 10^{15}, B - A \leqslant 3 \cdot 10^7$

### Input

Two integers $A$ and $B$.

### Output

Number of primes between them.

### Example

| countpr.in | countpr.out |
|---|---|
| 5 10 | 2 |
| 20 50 | 7 |

## Problem C. H. Reversive Inversions II

| | |
|---|---|
| Input file: | `invers.in` |
| Output file: | `invers.out` |
| Time limit: | 2 seconds |
| Memory limit: | 256 mebibytes |

*Inversion table* for a permutation $P$ of numbers $\{1, 2, \ldots, N\}$ is the table $A = (A_i)_{1 \leq i \leq N}$ which maps each $i = P_j$ into the number of indices $j'$ such that $j' \leq j$ but $P_{j'} > P_j = i$.

Given an inversion table for a permutation $P$, calculate the inversion table for the inverse permutation $P^{-1}$.

### Input

File consists only of $N$ integer numbers, delimited by spaces and newline characters, that form the inversion table of a permutation. You may assume that $1 \leq N \leq 262144$.

### Output

Output $N$ integer numbers separated by single spaces — inversion table for the inverse permutation. Leave no trailing spaces at the end of the single line of output.

If there are several possible answers, output any of them. If there are no answers, output the first $N$ primes instead.

### Example

| invers.in | invers.out |
|---|---|
| 5 0 1 3 2 1 0 | 1 5 1 3 2 0 0 |

## Problem D. Martian King

| | |
|---|---|
| Input file: | `king.in` |
| Output file: | `king.out` |
| Time limit: | 2 seconds |
| Memory limit: | 64 megabytes |

There is a small Kingdom on Mars. There are some cities in the Kingdom, some of them are regional centers. The Kingdom has also a capital which may be a regional center or not. Cities are connected with one-way roads. Some of the roads are red and the remaining ones are blue. There is no more than one road of each color getting out of each city.

There is a tradition that the King goes to a trip each year. The trip starts from the capital city and consists of $L$ roads. The trip must end in some regional center. In historical-aesthetic purposes, the sequence of colors of visited roads is written in Chronicles. The red road is denoted by character '`0`', and the blue road is denoted by '`1`', so the result is a sequence of $L$ binary digits.

Young Martian Vasja studies history. Recently he found some parts of Chronicles describing the period of ruling of the King Ares. Vasja discovered the following requirement that should have been satisfied by the King. Each year the King must choose a new route, the sequence of which is greater than the corresponding one for previous year. When comparing, sequences are interpreted as binary numbers with leading zeros allowed. In the first year of his ruling the King was allowed to choose any route he wished. If the King is unable to find the route, he must finish his ruling. Vasja also discovered that Ares was very smart and he always chose such routes that maximized the length of his ruling period.

Since Vasja has only parts of Chronicles, he cannot answer some questions. For example:

- Which was King's route in $K$-th year of his ruling (years of ruling are enumerated starting from 1)?
- In which year Ares chose a route denoted by a given sequence $S$?
- Which route he chose after the route denoted by a given sequence $S$?
- Which route was chosen before the route denoted by a sequence $S$?

Vasja is not so smart as Ares so he asks you to answer the questions mentioned. Write a program that answers them.

### Input

The first line of the input contains five integers $N$, $M$, $F$, $L$, $Q$ — the number of cities, roads, regional centers; the length of King's route and the number of Vasja's questions ($1 \leqslant N \leqslant 50$, $1 \leqslant M \leqslant 100$,

$1 \leqslant F \leqslant N$, $1 \leqslant L \leqslant 60$, $1 \leqslant Q \leqslant 10\,000$). The cities are enumerated with integers from 1 to $N$. The capital has number 1.

The following $M$ lines describe the road system — three numbers in each line: $A_i$, $B_i$ and $C_i$. Their meaning is that the $i$-th road is going from city number $A_i$ to city number $B_i$, and its color is $C_i$ ('0', if the road is red, '1', if it is blue).

The next line contains $F$ different integers — the numbers of cities, that are regional centers.

The last $Q$ lines contain one question each. The format of question description is the following:

- ? $K$ — which route was chosen by the King in $K$-th year of his ruling ($1 \leqslant K \leqslant 5 \cdot 10^{18}$)?
- ! $S$ — which year the King chose the route denoted by $S$?
- > $S$ — which route Ares chose after the route denoted by $S$?
- < $S$ — which route was chosen before the route denoted by $S$?

The route is denoted by a sequence of $L$ characters '0' and '1'.

It is guaranteed that all questions are correct and the answer always exists.

### Output

Output file must contain $T$ lines, one for each question. For question ! $S$ it is required to output a single decimal integer. For all other questions the line must consist of $L$ characters '0' and '1' — the record corresponding to the route.

### Examples

| king.in | king.out |
|---|---|
| 1 2 1 3 2<br>1 1 0<br>1 1 1<br>1<br>? 3<br>! 111 | 010<br>8 |
| 2 3 2 5 4<br>1 1 0<br>1 2 1<br>2 1 0<br>2 1<br>? 5<br>! 00010<br>> 01010<br>< 01000 | 00101<br>3<br>10000<br>00101 |
| 4 6 1 6 5<br>1 2 0<br>2 3 0<br>3 4 0<br>4 3 1<br>3 2 1<br>2 1 1<br>1<br>? 3<br>? 4<br>> 000111<br>< 001101<br>! 010101 | 001101<br>010011<br>001011<br>001011<br>5 |

## Problem E. Next, Please...

| | |
|---|---|
| Input file: | next.in |
| Output file: | next.out |
| Time limit: | 2 seconds |
| Memory limit: | 256 megabytes |

There was a unique device in the Research Institute of Next Generation (RING). It generates ... well, it generated the next number partition of an integer number $n$ in lexicographical order. If the given partition was the last partition of $n$, it generated the first partition of $n + 1$.

Unfortunately, somebody recently tried to investigate shockproof characteristics of the device. But after this device was thrown out from the 10th floor window, something strange happened to it. For example, at 10:02 it generated 5 from 3 2. At 10:03 it generated 6 5 from 5 3 3. The scientists of RING think that the device now considers only partitions consisting only of numbers that are not less than some integer $m$, for example, minutes count.

Vasya's task is to help to check this hypothesis. He needs to write a program for simulating proposed new behavior of the device. More formally, his program should consider an infinite sequence $S(m)$ of number partitions which looks like $P(m)$ $P(m + 1)$ $P(m + 2)$ ..., where $P(t)$ is lexicographically ordered sequence of all partitions of $t$ consisting of numbers which are not less than $m$. Here a partition is defined as $t = a_1 + a_2 + \ldots a_k$, where $k \geqslant 1$, and $a_1 \geqslant a_2 \geqslant \ldots \geqslant a_k$. Lexicographical order is defined by saying that the first non-equal elements at position $p$ of two partitions $a$ and $b$ (where $b$ occurs later than $a$ in $P(t)$) must satisfy $a_p < b_p$. For example, the start of this sequence for $m = 2$ looks as follows: 2, 3, 2 2, 4, 3 2, 5, 2 2 2, 3 3, 4 2, 6, ...

The task of the program is, given $m$ and an element of $S(m)$, to generate the next one.

### Input

The first line of input contains two integers $k$ and $m$ ($1 \leqslant k \leqslant 50\,000$, $1 \leqslant m \leqslant 50\,000$). The second line consists of $k$ integers $a_i$ ($m \leqslant a_i \leqslant 10^6$). It is guaranteed that $a_1 \geqslant a_2 \geqslant \ldots \geqslant a_k$.

### Output

The first line should contain the length of the next element $k'$. The second line contains $k'$ numbers forming a non-increasing sequence — the next element. It is guaranteed that $k'$ will never exceed 50 000.

### Examples

| next.in | next.out |
|---|---|
| 2 2<br>3 2 | 1<br>5 |
| 3 3<br>5 3 3 | 2<br>6 5 |
| 1 2<br>5 | 3<br>2 2 2 |

## Problem F. Painter

| | |
|---|---|
| Input file: | painter.in |
| Output file: | painter.out |
| Time limit: | 4 seconds |
| Memory limit: | 256 mebibytes |

Your are given a totally white infinite straight line. A painter can select a segment on this line and paint it either black or white. After each operation, you need to find, what is the total number of black segments on the line, and their total length (note that black segments may merge or be split by a white segment).

### Input

The first line contains the total number of drawn segments ($1 \leqslant N \leqslant 100\,000$). $N$ lines follow, each describing one operation. Each operation is described with a line $c\ x\ l$, where $c$ is the color (`W` stands for white, `B` for black), and the segement drawn has coordinates $[x; x + l]$ (inclusive), both coordinates are integers that do not exceed 500 000 by an absolute value, so length is a positive integer up to $10^6$.

### Output

After each draw operation you must output on a separate line the number of black segments and their total length.

### Example

| painter.in | painter.out |
|---|---|
| 7<br>W 2 3<br>B 2 2<br>B 4 2<br>B 3 2<br>B 7 2<br>W 3 1<br>W 0 10 | 0 0<br>1 2<br>1 4<br>1 4<br>2 6<br>3 5<br>0 0 |

## Problem G. Permuatations

| | |
|---|---|
| Input file: | `permutation.in` |
| Output file: | `permutation.out` |
| Time limit: | 2 seconds |
| Memory limit: | 256 mebibytes |

You are given a permutation $P_i$ of the numbers $1..N$. Your task is to write a program which will answer the queries: for $x \leqslant j \leqslant y$, how many $P_j$ are between $k$ and $l$, inclusive?

### Input

First line contains two integers: $1 \leqslant N \leqslant 100\,000$ — the size of the permutation and $1 \leqslant M \leqslant 100\,000$ — the number of queries. The second line contains $N$ numbers — the permutation itself. $M$ lines follow, one for each query. Each of them contains four integers: $1 \leqslant x \leqslant y \leqslant N$ and $1 \leqslant k \leqslant l \leqslant N$, in that order.

### Output

$M$ lines — answers for the queries, in order they appeared in the input.

### Example

| permutation.in | permutation.out |
|---|---|
| 4 2<br>1 2 3 4<br>1 2 2 3<br>1 3 1 3 | 1<br>3 |

## Problem H. Primes in a Sequence

| | |
|---|---|
| Input file: | `primes.in` |
| Output file: | `primes.out` |
| Time limit: | 2 seconds |
| Memory limit: | 256 megabytes |

R.I.P. (Research Institute of Primes) calls for help! Vasya is here today to help them to search for primes in very specific sequences. These sequences start with an arbitrary positive integer $a_1$. The following numbers of the sequence are evaluated using formula $a_{i+1} = d(a_i) + i$. Here $d(x)$ stands for the number of divisors of $x$. For example, if the sequence starts from $a_1 = 2^{32\,582\,657} - 1$, then $a_2 = 3$, $a_3 = 4$ and $a_4 = 6$.

The $(l, r)$-*window* of some sequence is defined as the subsequence of this sequence between its $l$-th and $r$-th element, inclusive.

Vasya's task now is to determine maximal possible number of primes (each occurence of any repeated prime is accounted) in $(l, r)$-window over all sequences of such kind for given $l$ and $r$. The problem is too hard for him because there is an infinite number of such sequences. For example, the sequence described above is an example of optimal sequence for $(1, 4)$-window. So he asked you for help.

### Input

The input consists of no more than one thousand cases. Each case is a single line with two integers $l$ and $r$ ($1 \leqslant l < r \leqslant 10^6$). Input is terminated by the line of two zeroes. The total sum of $l + r$'s over all cases in a single input does not exceed $10^7$.

### Output

Output the maximal possible number of primes as shown below.

### Examples

| primes.in |
|---|
| 1 4<br>6 7<br>10 20<br>0 0 |
| **primes.out** |
| Window 1: There can be up to 2 primes.<br>Window 2: There can be up to 1 primes.<br>Window 3: There can be up to 4 primes. |

## Problem I. Josephus Problem-2

| | |
|---|---|
| Input file: | `psyche.in` |
| Output file: | `psyche.out` |
| Time limit: | 2 seconds |
| Memory limit: | 64 megabytes |

$n$ boys are standing in circle. They start counting themselves clockwise, starting from 1. As soon as the count reaches $p$, the last boy counted leaves the circle, and they continue counting from the next boy, starting from 1 again.

Last remaining boy wins.

Can you calculate his number in clockwise order, if the boy from whom the counting originally started has number 1?

### Input

Input file contains two integer numbers, $n$ and $p$. ($1 \leq n \leqslant 10^{18}$, $1 \leqslant p \leq 1000$).

### Output

Output file should contain one number — the original number of the last boy.

### Example

| psyche.in | psyche.out |
|---|---|
| 5 3 | 4 |

## Problem J. Stars

| | |
|---|---|
| Input file: | `stars.in` |
| Output file: | `stars.out` |
| Time limit: | 4 seconds |
| Memory limit: | 256 mebibytes |

The space cube of size $n \times n \times n$ is divided to small cubes of size $1 \times 1 \times 1$. The following events may occur:

1. In one of the small cubes a number of stars appear or disappear.
2. Somebody asks how many stars does contain a given rectangular parallelipiped with sides parallel to coordinate axes.

### Input

The first line of the input file contains an integer $1 \leqslant n \leqslant 128$. The $x$, $y$ and $z$ coordinates of small cubes are integers in range $0..n-1$. The list of events follows. Each event is listed on a separate line. Line starts from an integer $m$ which can be one of the following:

1. $m = 1$ followed by 4 integers — $x$, $y$, $z$ ($0 \leqslant x, y, z < N$) and $k$ ($-20000 \leqslant k \leqslant 20000$) — which are coordinates of the small cube and the number of stars appeared or disappeared in this small cube (positive if some stars appeared, negative otherwise, can be zero if there are no changes).
2. $m = 2$ is followed by 6 integers — $x_1$, $y_1$, $z_1$, $x_2$, $y_2$, $z_2$ ($0 \leqslant x_1 \leqslant x_2 < N$, $0 \leqslant y_1 \leqslant y_2 < N$, $0 \leqslant z_1 \leqslant z_2 < N$), a query to calculate the total number of stars $(x, y, z)$ in area containing all the small cubes satisfying: $x_1 \leqslant x \leqslant x_2$, $y_1 \leqslant y \leqslant y_2$, $z_1 \leqslant z \leqslant z_2$;
3. $m = 3$ — stop working, it will be always the last event of the input

The number of events in the input does not exceed 100 002.

### Output

For each query ($m = 2$) output the answer on a separate line.

### Example

| stars.in | stars.out |
|---|---|
| 2<br>2 1 1 1 1 1 1<br>1 0 0 0 1<br>1 0 1 0 3<br>2 0 0 0 0 0 0<br>2 0 0 0 0 1 0<br>1 0 1 0 -2<br>2 0 0 0 1 1 1<br>3 | 0<br>1<br>4<br>2 |

## Problem K. Cartesian Tree

| | |
|---|---|
| Input file: | `tree.in` |
| Output file: | `tree.out` |
| Time limit: | 2 seconds |
| Memory limit: | 64 megabytes |

Let us consider a special type of binary search trees, called *cartesian trees*. Recall that a binary search tree is a rooted ordered binary tree, such that for its every node $x$ the following condition is satisfied: each node in its left subtree has the key less than the key of $x$, and each node in its right subtree has the key greater than the key of $x$.

That is, if we denote the left subtree of the node $x$ by $L(x)$, its right subtree by $R(x)$ and its key by $k_x$, for each node $x$ we will have

- if $y \in L(x)$ then $k_y < k_x$
- if $z \in R(x)$ then $k_z > k_x$

The binary search tree is called *cartesian* if its every node $x$ in addition to the main key $k_x$ also has an auxiliary key that we will denote by $a_x$, and for these keys the *heap condition* is satisfied, that is

- if $y$ is the parent of $x$ then $a_y < a_x$

Thus a cartesian tree is a binary rooted ordered tree, such that each of its nodes has a pair of two keys $(k, a)$ and three conditions described are satisfied.

Given a set of pairs, construct a cartesian tree out of them, or detect that it is not possible.

### Input

The first line of the input file contains an integer number $N$ — the number of pairs you should build cartesian tree out of ($1 \leq N \leq 50\,000$). The following $N$ lines contain two integer numbers each — given pairs $(k_i, a_i)$. For each pair $|k_i|, |a_i| \leq 30\,000$. All main keys and all auxiliary keys are different, i.e. $k_i \neq k_j$ and $a_i \neq a_j$ for each $i \neq j$.

### Output

On the first line of the output file print `YES` if it is possible to build a cartesian tree out of given pairs or `NO` if it is not. If the answer is positive, output the tree itself in the following $N$ lines. Let the nodes be numbered from 1 to $N$ corresponding to pairs they contain as these pairs are given in the input file. For each node output three numbers — its parent, its left child and its right child. If the node has no parent or no corresponding child, output `0` instead.

If there are several possible trees, output any one.

### Example

| tree.in | tree.out |
|---|---|
| 7<br>5 4<br>2 2<br>3 9<br>0 5<br>1 3<br>6 6<br>4 11 | YES<br>2 3 6<br>0 5 1<br>1 0 7<br>5 0 0<br>2 4 0<br>1 0 0<br>3 0 0 |