

## Problem A. A Dirty Tricks

Input file:            **standard input**  
Output file:         **standard output**  
Time limit:          1 second  
Memory limit:       1024 megabytes

The famous magician Bill has developed a new trick.

The spectator names an integer  $n$  from 1 to  $10^7$  inclusive, after which Bill instantly states the remainder of the product of the first  $n$  prime numbers when divided by  $10^{10} + 9$ .

The circus management wants to hire Bill and has tasked you with writing a program to verify his answers.

### Input

The input contains a single integer  $n$  ( $1 \leq n \leq 10^7$ ).

### Output

Output a single integer — the remainder of the product of the first  $n$  prime numbers when divided by  $10^{10} + 9$ .

### Examples

| standard input | standard output |
|----------------|-----------------|
| 1              | 2               |
| 2              | 6               |

## Problem B. Big Sieve Game

Input file: *standard input*  
Output file: *standard output*  
Time limit: 1 second  
Memory limit: 256 mebibytes

After mastering the sieve of Eratosthenes, Alice excitedly created a puzzle game that made use of it. The rules of the puzzle game are as follows:

- An array  $p_1, p_2, \dots, p_n$  is given. Initially, all  $p_i$  are 0.
- A target array  $t_1, t_2, \dots, t_n$  is given.
- The player can perform the following two types of operations:
  - Choose  $i$  and increase  $p_j$  by 1 for every  $j$  divisible by  $i$ .
  - Choose  $i$  and decrease  $p_j$  by 1 for every  $j$  divisible by  $i$ .
- The puzzle is solved when  $p = t$  after applying zero or more operations.

Alice aims to solve the puzzle using the fewest operations, showcasing her puzzle-solving skills. Please help Alice find the minimum number of operations to solve the puzzle.

### Input

The first line contains a single integer  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ).

The second line contains  $n$  space-separated integers  $t_i$ : the elements of the array  $t$  ( $-10^9 \leq t_i \leq 10^9$ ).

### Output

Print the minimum number of operations to solve the puzzle. If the puzzle is unsolvable, print -1.

### Examples

| <i>standard input</i> | <i>standard output</i> |
|-----------------------|------------------------|
| 4<br>1 1 1 0          | 2                      |
| 7<br>0 1 1 1 0 2 -1   | 3                      |

## Problem C. Catch The Flea

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 1024 mebibytes

You have placed glues on each cell of an  $n \times m$  grid to create a rectangular flea trap. Each glue has a *weak direction*; if a flea on the glue jumps towards its *weak direction*, the flea can jump out of the glue.

More precisely, each glue is represented by U, D, L, or R, meaning up, down, left, and right respectively.

In a single jump, a flea can move any distance up to  $k$  cells in the weak direction. If a flea jumps out of the rectangle, we say that the flea has *escaped*.

You became curious about how efficient your trap is. If a flea that is placed on a cell of the trap can escape after consecutive jumps, we call the cell an *escapable cell*. Your task is to count the number of escapable cells.

### Input

The first line of input contains three integers: the trap sizes  $n$  and  $m$  and the jump limit  $k$  ( $1 \leq n, m, k \leq 2000$ ).

Each of the next  $n$  lines contains a string of length  $m$  consisting of letters U, D, L, and R. These lines indicate the *weak direction* of each glue.

### Output

Print the number of escapable cells.

### Example

| <i>standard input</i>                              | <i>standard output</i> |
|--|------------------------|
| 5 5 2<br>DDDRD<br>DDDDD<br>RDLUL<br>UURJU<br>UUUUU | 14                     |

### Note

Fleas which start from (1,3), (1,4), (1,5), (2,3), (2,4), (2,5), (3,4), (3,5), (4,3), (4,4), (4,5), (5,3), (5,4), and (5,5) can escape. For example:

- A flea at (1,3) can escape by jumping along the following path:  
(1,3)  $\rightarrow$  (2,3)  $\rightarrow$  (4,3)  $\rightarrow$  (4,4)  $\rightarrow$  (3,4)  $\rightarrow$  (1,4)  $\rightarrow$  out of the trap
- A flea at (1,4) can escape by jumping two cells to the right.

## Problem D. Dual Language Stable Triples

Input file:            **standard input**  
Output file:          **standard output**  
Time limit:          1 second  
Memory limit:        256 megabytes

In Pascal,  $\wedge$  is exponentiation, while in C++ it is XOR (the bitwise logical operation “exclusive or”).

We call a triple of non-negative integers  $a$ ,  $b$ , and  $c$  *stable* if  $a + b > 0$  and  $a \wedge b = c$  in both Pascal and C++.

Determine for given integers  $x$  and  $y$ , how many distinct stable triples  $a$ ,  $b$ ,  $c$  exist such that  $x \leq a, b, c \leq y$ . Triples that differ by the permutation of numbers are considered distinct (for example, the triples (29, 9, 24) and (9, 29, 24) are considered different).

### Input

The first line of input contains one integer  $x$  — the lower bound of the range ( $0 \leq x \leq 10^6$ ). The second line of input contains one integer  $y$  — the upper bound of the range ( $x \leq y \leq 10^6$ ).

### Output

Output one integer — the number of stable triples between  $x$  and  $y$ , inclusive.

### Examples

| standard input | standard output |
|----------------|-----------------|
| 0<br>1         | 1               |
| 1<br>1         | 0               |

## Problem E. Excellent HLD

Input file: *standard input*  
Output file: *standard output*  
Time limit: 4 seconds  
Memory limit: 256 mebibytes

You are given a rooted tree consisting of  $n$  vertices. Its root is vertex 1. Let us consider a *heavy-light decomposition* of the tree, where each edge is either a *heavy edge* or a *light edge*. For each vertex, among all edges connecting the vertex with its children, at most one edge can be a heavy edge.

In this problem, we have a multiset of simple paths  $T$ , which is initially empty. We will assign each edge to be a heavy edge or a light edge according to  $T$ , satisfying the condition above.

Each time an update is done on  $T$ , your task is to find an assignment of edges that minimizes the sum of the number of light edges on every path in  $T$ .

There are  $q$  updates given. Each update consists of three integers:  $s$ ,  $e$ , and  $k$ . They mean that  $k$  copies of the simple path from  $s$  to  $e$  are inserted into  $T$ . After each update, find the minimum sum of the number of light edges on every path in  $T$ .

### Input

The first line of input contains two space-separated integers  $n$  and  $q$  ( $2 \leq n \leq 10^5$ ;  $1 \leq q \leq 10^5$ ).

The  $i$ -th of the following  $n - 1$  lines contains two space-separated integers  $x_i$  and  $y_i$ , meaning that the  $i$ -th edge connects vertices  $x_i$  and  $y_i$  in the tree ( $1 \leq x_i, y_i \leq n$ ;  $x_i \neq y_i$ ). It is guaranteed that the given edges form a tree.

The  $i$ -th of the following  $q$  lines contains three space-separated integers,  $s$ ,  $e$ , and  $k$ , describing each update ( $1 \leq s, e \leq n$ ;  $s \neq e$ ;  $1 \leq k \leq 10^9$ ).

The updates are processed in the input order. The updates are permanent: the changes made in each update persist in further updates as well.

### Output

For each of the  $q$  updates, print a line with the answer after this update.

## Examples

| <i>standard input</i>  | <i>standard output</i>                    |
|--|---|
| 3 3<br>1 2<br>3 1<br>1 3 2<br>1 3 3<br>1 2 2   | 0<br>0<br>2                               |
| 5 5<br>3 4<br>2 4<br>1 2<br>5 3<br>5 4 2<br>1 2 4<br>3 4 1<br>5 3 4<br>1 2 2   | 0<br>0<br>0<br>0<br>0                     |
| 8 8<br>4 6<br>8 4<br>1 6<br>5 1<br>2 1<br>3 2<br>7 3<br>2 7 1<br>8 2 1<br>5 3 6<br>8 3 5<br>1 4 2<br>6 7 1<br>5 6 4<br>6 2 3 | 0<br>1<br>7<br>12<br>14<br>15<br>23<br>26 |

## Problem F. Full Irreducibility

Input file: *standard input*  
Output file: *standard output*  
Time limit: 1 second  
Memory limit: 256 mebibytes

A permutation is called *irreducible* if none of its prefixes form a permutation, except the full permutation itself. For example,  $[2, 3, 1]$  and  $[4, 1, 2, 3]$  are irreducible, while  $[2, 1, 3]$  and  $[1, 3, 2]$  are not.

You are given a permutation  $p$  of length  $n$ . In one operation, you can choose any two adjacent positions and swap the values at these positions.

Find the minimum number of operations to transform  $p$  into an irreducible permutation, and the corresponding sequence of operations itself.

### Input

The first line of input contains a single integer  $t$ , the number of test cases ( $1 \leq t \leq 10^5$ ).

The first line of each test case contains a single integer  $n$  ( $1 \leq n \leq 10^5$ ).

The second line of each test case contains  $n$  integers  $p_1, \dots, p_n$  ( $1 \leq p_i \leq n$ ).

It is guaranteed that the sum of  $n$  over all test cases does not exceed  $5 \cdot 10^5$ .

### Output

For each test case, print two lines:

On the first line, print an integer  $k$ : the minimum number of operations that make  $p$  irreducible.

On the second line, print  $k$  space-separated integers  $s_1, \dots, s_k$  ( $1 \leq s_i \leq n - 1$ ) where  $s_i$  and  $s_i + 1$  are the positions of the values you intend to swap. The operations are performed sequentially in the order specified by your output. If there are multiple solutions, print any one of them.

### Example

| <i>standard input</i> | <i>standard output</i> |
|-----------------------|------------------------|
| 3                     | 3                      |
| 4                     | 1 2 3                  |
| 1 2 3 4               | 0                      |
| 3                     |                        |
| 2 3 1                 | 2                      |
| 5                     | 4 3                    |
| 3 1 2 4 5             |                        |

## Problem G. Good Triangle

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 1024 mebibytes

You are given  $n$  distinct points on a two-dimensional plane.

We define the distance between two points  $P = (x_1, y_1)$  and  $Q = (x_2, y_2)$  as  $d(P, Q) = |x_1 - x_2| + |y_1 - y_2|$ .

Let us say that three distinct points  $U, V, W$  form a *good triangle* if there exists a point  $T$  such that  $d(U, T) = d(V, T) = d(W, T)$ . Note that  $T$  does not have to be a lattice point.

Find the number of good triangles that can be formed by the given points.

### Input

The first line of the input contains one integer  $N$  ( $3 \leq N \leq 5 \cdot 10^5$ ).

The  $i$ -th of the next  $N$  lines contains two space-separated integers  $x_i$  and  $y_i$ , meaning that the coordinate of the  $i$ -th point is  $(x_i, y_i)$  ( $-10^9 \leq x_i, y_i \leq 10^9$ ;  $(x_i, y_i) \neq (x_j, y_j)$  if  $i \neq j$ ).

### Output

Print one integer: the number of good triangles that can be formed by the given points.

### Examples

| <i>standard input</i>  | <i>standard output</i> |
|--|------------------------|
| 5<br>1 -1<br>1 5<br>5 7<br>1 3<br>4 2  | 9                      |
| 10<br>-2 -1<br>-2 2<br>-1 -2<br>-1 -1<br>-1 1<br>0 1<br>1 -1<br>1 2<br>2 -1<br>2 1 | 108                    |



## Problem H. How to Guess Power of Two

Input file:            **standard input**  
Output file:          **standard output**  
Time limit:           **1 second**  
Memory limit:        **1024 megabytes**

This is an interactive problem.

An integer  $x$  from 0 to 63 is chosen. You can ask the jury program the following queries: add an integer  $d$  from  $-3$  to  $3$  to the chosen number and output the sum of the digits in the decimal representation of the number  $2^{x+d}$ . However, if  $x + d$  is less than 0 or greater than 63, you lose.

Your task is to guess the number  $x$  in two attempts.

### Interaction Protocol

The interaction starts with your program outputting a query in the form of “?  $d$ ”, where  $-3 \leq d \leq 3$  is an integer.

If  $x + d$  goes out of the range from 0 to 63 inclusive, your solution is immediately deemed incorrect. Otherwise, the program outputs the answer to the query—the sum of the digits in the decimal representation of the number  $2^{x+d}$ . You can make no more than two queries.

To output the answer, use the format “!  $x$ ”. Outputting the answer does not count as a query.

Remember to output a newline character after each query or answer output.

### Example

| standard input | standard output |
|----------------|-----------------|
| 2              | ? -1            |
| 8              | ? 1             |
|                | ! 2             |

## Problem I. Isn't It Beautiful?

Input file: *standard input*  
Output file: *standard output*  
Time limit: 1 second  
Memory limit: 256 mebibytes

We define the beauty of an array as its minimum excluded value (MEX): the smallest non-negative integer that does not belong to the array. A larger MEX corresponds to a more beautiful array.

You are given an array  $a$  of length  $n$ . You want to enhance it in terms of its beauty. To achieve this, you can choose a non-negative integer  $x$  and replace each element  $a_i$  with  $a_i \& x$ . Here,  $\&$  denotes the bitwise AND operator: each bit of the result is equal to the logical AND of the respective bits of the operands.

Find the value of  $x$  that maximizes the beauty of the array.

### Input

The first line of input contains a single integer  $t$ , the number of test cases ( $1 \leq t \leq 10^5$ ).

Each test case is given on two lines.

The first of these lines contains a single integer  $n$  ( $1 \leq n \leq 10^5$ ).

The second line contains  $n$  integers  $a_1, \dots, a_n$  ( $0 \leq a_i < 2^{30}$ ).

It is guaranteed that the sum of  $n$  over all test cases does not exceed  $10^5$ .

### Output

For each test case, print a single integer  $x$  that maximizes the MEX of the array formed by taking the bitwise AND of each element of  $a$  with  $x$ . This value should satisfy  $0 \leq x < 2^{30}$ ; it can be shown that there always exists an optimal  $x$  in that range. If there are multiple solutions, print any one of them.

### Example

| <i>standard input</i>       | <i>standard output</i> |
|-----------------------------|------------------------|
| 1<br>6<br>13 11 40 10 33 19 | 23                     |

### Note

In the sample, we can choose  $x = 23$ , so the new array will be

$$[13\&23, 11\&23, 40\&23, 10\&23, 33\&23, 19\&23],$$

which is  $[5, 3, 0, 2, 1, 19]$  with a MEX of 4. Another possible answer is  $x = 19$ .

## Problem J. Just an ACM-subsequence

Input file:           standard input  
Output file:         standard output  
Time limit:          1 second  
Memory limit:       1024 megabytes

We will call a string an *ACM-string* if we can remove all characters except for three such that the remaining three letters form the word **ACM**.

Given a string consisting of lowercase Latin letters, constructed by a random uniform selection among all strings of length  $n$ . What is the probability that this string is an ACM-string?

### Input

The first line of the input contains a single integer  $n$  — the length of the string ( $1 \leq n \leq 10^5$ ).

### Output

It can be shown that the probability can be represented in a unique way as a fraction  $p/q$ , where  $p$  and  $q$  are coprime, and the denominator is coprime with 998 244 353. Output the value of  $p \cdot q^{-1}$  modulo 998 244 353, that is, such a number  $0 \leq x \leq 998\,244\,353$ , that  $q \cdot x$  gives the same remainder when divided by 998 244 353 as  $p$ .

### Examples

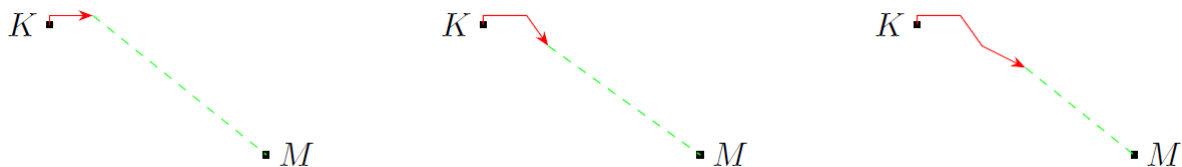
| standard input | standard output |
|----------------|-----------------|
| 4              | 448903698       |
| 2              | 0               |

## Problem K. Kingdom And Airlines

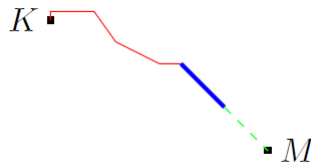
Input file:            **standard input**  
Output file:          **standard output**  
Time limit:           **1 second**  
Memory limit:        **1024 megabytes**

The flight route of the airplane from the Kingyo, the capital of the some kingdom on the islands to the city Kingoto on the far island is defined by  $n$  checkpoints (the first of which is Kingyo, the last is city Kingoto, and each subsequent checkpoint is strictly eastern of the previous one). Between the checkpoints, the airplane flies in a straight line. So the trajectory of the flight from Kingyo to Kingoto is represented by a rather winding polyline.

On the map of the flight displayed on the media center screens, the already traversed trajectory of the airplane is marked with a solid red line, while the current direction in a straight line towards Kingoto is marked with a green dashed line. In other words, the green segment is the line segment connecting the current position of the airplane and city Kingoto.



We will consider that the airplane is flying along a *estimated direction* if the green segment is a continuation of the segment of the polyline along which the airplane is currently moving.



Given the checkpoints, determine the distance the airplane has flown along a estimated direction.

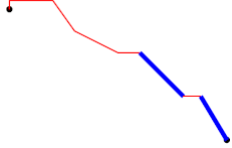
### Input

The first line of input contains a single integer  $n$  ( $2 \leq n \leq 1000$ ) — the number of checkpoints. Each of the following  $n$  lines contains two integers  $x_i$  and  $y_i$  — the coordinates of the next checkpoint in the order of the route. The first point corresponds to city Kingyo, and the last point corresponds to Kingoto. ( $-10^4 \leq x_i, y_i \leq 10^4$ ,  $x_i > x_j$  for  $i > j$ ). Note that three consecutive checkpoints may lie on the same straight line.

### Output

Output the distance that the airplane has flown along a estimated direction, with an absolute or relative error no worse than  $10^{-4}$ .

Example

| standard input   | standard output   | Notes   |
|--|-------------------|---|
| 10<br>-2 -2<br>-1 0<br>4 0<br>8 0<br>13 -7<br>23 -12<br>28 -12<br>38 -22<br>42 -22<br>48 -32 | 25.80403941342155 |  <p>Segments of flight along a smooth trajectory are highlighted: from (28,-12) to (38,-22) and from (42,-22) to city M. (48,-32). Their total length is equal to <math>\sqrt{200} + \sqrt{136} \approx 25.80404</math>.</p> |

## Problem L. Lottery

Input file: *standard input*  
Output file: *standard output*  
Time limit: 1 second  
Memory limit: 256 mebibytes

You are a big fan of Korea's biggest rockstar, Koosaga. Excitingly, Koosaga has announced a lottery event offering fans a once-in-a-lifetime opportunity for a one-on-one meeting.

Koosaga sells  $n$  kinds of albums. When you purchase the  $i$ -th kind of album, it costs you  $a_i$  won, and you get  $b_i$  lottery tickets. You can purchase multiple copies of the same album if you wish.

On the day of the draw, a large roulette wheel containing cells with participant names will decide the winner. Every cell on the wheel has an equal chance of being selected. The number of cells with your name corresponds to the number of lottery tickets you've amassed.

Koosaga will give the wheel a single spin to pick the winner. But if luck isn't on your side initially, there's still hope! By paying  $r$  won, you can request Koosaga to spin the wheel again. You can pay for as many re-spins as you desire.

Armed with insider knowledge, you've learned that the cumulative number of cells attributed to other participants is  $s$ . Importantly, none of them will opt for a re-spin.

Your challenge is to find an optimal strategy to win with the minimum expected cost.

### Input

The first line contains three integers:  $n$ ,  $s$ , and  $r$  ( $1 \leq n \leq 10^5$ ;  $1 \leq s \leq 10^6$ ;  $1 \leq r \leq 10^6$ ). The  $i$ -th of the next  $n$  lines contains two integers:  $a_i$  and  $b_i$  ( $1 \leq a_i \leq 300$ ;  $1 \leq b_i \leq 5000$ ).

### Output

Print a single line with two positive integers  $x$  and  $y$  which must be coprime. The value  $\frac{x}{y}$  must be the minimum expected cost to win. It is guaranteed that the minimum expected cost can be expressed in this format.

### Example

| <i>standard input</i>        | <i>standard output</i> |
|------------------------------|------------------------|
| 3 11 3<br>1 3<br>2 7<br>5 13 | 63 10                  |

## Problem M. Make It Regular

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 1024 mebibytes

You are given a bracket sequence consisting of  $n$  opening brackets and  $n$  closing brackets. Let  $S$  be a **non-empty** set of integers between 1 and  $2n$ , inclusive. You can choose two indices in  $S$ , not necessarily adjacent, and swap the brackets of the bracket sequence at those two positions.

Find the number of  $S$  that make it possible to obtain a regular bracket sequence by repeatedly applying this operation an arbitrary number of times. As this number may be very large, find it modulo the prime number 998 244 353.

### Input

The first line contains one integer  $n$  ( $1 \leq n \leq 3000$ ).

The second line contains a string of  $2n$  brackets, either "(" or ")". The given bracket sequence contains  $n$  opening brackets and  $n$  closing brackets.

### Output

Print the number of all possible  $S$  modulo 998 244 353.

### Examples

| <i>standard input</i> | <i>standard output</i> |
|-----------------------|------------------------|
| 3<br>( )) ( (         | 36                     |
| 6<br>( )) ( ( ) ( ( ( | 1536                   |