

Problem A. Balanced Arrays

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Misha is playing with an array of integers. In one operation, he may do one of the following:

- Add 1 to some suffix of the array.
- Add 1 to some prefix of the array.

For example, if Misha has the array $(1, 2, 4)$, in one operation, he can obtain one of the arrays $(2, 2, 4)$, $(2, 3, 4)$, $(2, 3, 5)$, $(1, 2, 5)$, or $(1, 3, 5)$.

An array of length n is called *balanced* if Misha can obtain it from an array of n zeroes after some operations. For example, the array $(1, 2, 1)$ is balanced, but the array $(1, 3, 1)$ is not. How many arrays of length n with elements at most m are balanced? The answer can be very large, so, output it modulo prime number 998 244 353.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 500\,000$).

Output

Output a single integer: the number of balanced arrays of length n with elements at most m . Print the answer modulo prime number 998 244 353.

Example

<i>standard input</i>	<i>standard output</i>
2 2	9

Problem B. Bins and Balls

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

You have several balls of n different colors. For each color i from 1 to n , there are exactly x_i balls of this color. You are playing a game which is a sequence of actions. In one action, you can take exactly k balls of pairwise distinct colors and throw them away. What is the maximum number of actions that you can make?

Input

The first line contains two integers n and k : the number of colors and the number of balls that you throw away in each action ($1 \leq k \leq n \leq 2 \cdot 10^5$). The second line contains n space-separated integers x_i : the number of balls of the i -th color ($1 \leq x_i \leq 10^9$).

Output

Print a single line with one integer: the maximum possible number of actions you can make.

Examples

<i>standard input</i>	<i>standard output</i>
4 3 5 8 9 4	8
10 5 1 2 3 4 5 6 239 239 239 239	21

Problem C. Cards

Input file: *standard input*
 Output file: *standard output*
 Time limit: 15 seconds
 Memory limit: 512 mebibytes

Nikita is playing tricks with cards. Each card has an integer between -2 and 2 , inclusive, written on it. Nikita has a magician's hat with cards, and also a secret number which is n initially. Nikita repeats the following operation m times: he takes a random card from the hat, adds the number written on the card to his secret number, and puts the card back into the hat.

If the secret number becomes negative, Nikita loses the game immediately. He wins if he didn't lose after m operations. What is the probability that he will win?

Input

The first line contains two integers: n and m ($0 \leq n, m \leq 100\,000$).

The second line contains five integers: x_{-2} , x_{-1} , x_0 , x_1 , and x_2 ($0 \leq x_i \leq 10^8$). Here, x_i is the number of cards with i written on it. There is at least one card in the hat.

Output

Print a single integer: the probability modulo the prime number 998 244 353. That is, the desired probability is a rational number $\frac{p}{q}$. You should output $p \cdot q^{-1}$ modulo 998 244 353.

Example

<i>standard input</i>	<i>standard output</i>
1 1 1 1 1 1 1	399297742

Note

In the first testcase probability is equal $\frac{4}{5}$.

Problem D. Fairy Chess

Input file: *standard input*
 Output file: *standard output*
 Time limit: 2 seconds
 Memory limit: 512 mebibytes

Alice and Bob have a set consisting of regular and fairy chess pieces. There are a total of 6 types of pieces in the set (bishop, rook, queen, archbishop, chancellor, and maharaja). There are exactly two pieces of each type in the set. A rook can move any number of squares along a rank or file. A bishop can move any number of squares diagonally. The queen combines the power of a rook and bishop and can move any number of squares along a rank, file, or diagonal. A knight moves to any of the closest squares that are not on the same rank, file, or diagonal. Thus the move forms an L-shape: two squares vertically and one square horizontally, or two squares horizontally and one square vertically. The archbishop moves like a bishop and a knight, the chancellor moves like a rook and a knight, and the maharaja moves like a queen and a knight.

Alice and Bob decided to play a game on an 8 by 8 chessboard. At the beginning of the game, the pieces are shuffled and placed in a row, establishing an order. Alice places the first piece on the board in the given order, Bob places the second piece, Alice places the third piece, and so on. Each piece must be placed on an empty square in such a way that it does not capture and is not captured by any previously placed piece. The player who is unable to make a move loses.

Your task is to write a program that determines who will win if both players play optimally.

Input

The input consists of a string consisting of 12 uppercase letters, representing the order in which the pieces are placed on the board. “B” corresponds to the bishop, “R” corresponds to the rook, “Q” corresponds to the queen, “A” corresponds to the archbishop, “C” corresponds to the chancellor, and “M” corresponds to the maharaja. There are exactly two pieces of each of the six types.

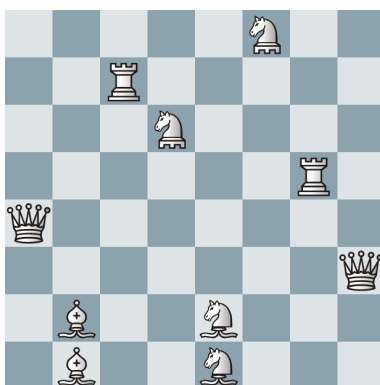
Output

Output “Alice” if Alice wins. Output “Bob” if Bob wins.

Examples

<i>standard input</i>	<i>standard output</i>
BBAARRCCQQMM	Bob
BAMBAMQQRCCR	Alice

Note



Here is a possible final position for the the first example if the opponents did not play optimally. The first ten pieces (all except maharajas) are on the board. The pieces at d6 and f8 are chancellors. The pieces at e1 and e2 are archbishops. The eleventh piece (maharaja) can not be placed.

Problem E. Fair Elections

Input file: *standard input*
 Output file: *standard output*
 Time limit: 4 seconds
 Memory limit: 1024 mebibytes

In the City of Truth, an election started. There are three candidates running for president, numbered from 1 to 3. Also, there are n voters. Every voter has their own list of preferences, set by a permutation of numbers 1, 2, 3. For example, permutation (2, 3, 1) means that this voter prefers candidate 2 the most, then they prefer candidate 3, and candidate 1 is the least preferable for them. Lists of preferences are known to everyone.

Voters will vote (that is, choose one candidate) one by one in fixed order, from the first voter to the last voter. Moreover, after a person votes, they will immediately and honestly tell how they voted.

The candidate who was chosen by most voters will be the president. If several candidates have the most votes, the winner is the candidate with the lowest number (for example, if the candidates 1 and 2 both have the most votes, then candidate 1 wins).

Who will win if every person votes optimally?

Input

The first line contains a single integer n : the number of voters ($1 \leq n \leq 10\,000$).

Then n lines follow. The i -th of these lines contains a permutation of numbers 1, 2, and 3: the preferences of the i -th voter.

Output

Output a single integer from 1 to 3: the number of the candidate who will win the election.

Example

<i>standard input</i>	<i>standard output</i>
3 3 2 1 1 2 3 2 1 3	2

Problem F. Exactly Three Neighbors

Input file: *standard input*
 Output file: *standard output*
 Time limit: 2 seconds
 Memory limit: 512 mebibytes

Consider a field of squares, infinite in all directions. Each square is painted either black or white. Each black square shares a side with **exactly three** black neighbors.

We will consider periodic colorings. More precisely, let us first color a rectangle of cells. Then divide the field into such rectangles, joining them by sides. The coloring will be the same in every rectangle.

Provide an example of a coloring where the total share of black squares is equal to the given rational number p/q , or determine that it is impossible.

Input

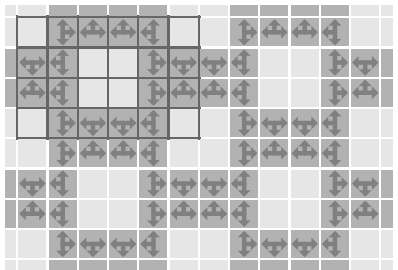
The first line contains two integers p and q : the numerator and denominator of the desired total share of black squares ($0 \leq p \leq q \leq 10$; numbers p and q are relatively prime).

Output

If the desired coloring is possible, on the first line, print two integers h and w : the height and width of the rectangle ($1 \leq h, w \leq 1000$). Then print the coloring of the rectangle consisting of h lines with w characters in each. Character “.” (dot) describes a white square, and character “#” (hash) describes a black square. The ratio of the number of black squares in the rectangle to the total number of squares in the rectangle should be p/q . If there are several possible colorings, print any one of them.

If the desired coloring is impossible, print “-1 -1” on the first line.

Examples

<i>standard input</i>	<i>standard output</i>	<i>illustration</i>
2 3	4 6 .####. ##.## ##.## .####.	
1 1	-1 -1	

Problem G. Lake

Input file: *standard input*
 Output file: *standard output*
 Time limit: 2 seconds
 Memory limit: 512 mebibytes

On a hot summer day, Tatiana and the children set off on foot to the lake. The journey was so exhausting that everyone was delighted when their friend Olga suddenly flew to them in a helicopter after they had finished swimming.

There are two rows of passenger seats available in the helicopter. To reduce the number of flights, Olga allowed an adult to sit with a child on all passenger seats except the center rear seat. And in the center rear row, either one adult or one child can sit.

Below is a diagram indicating where an adult can sit together with a child on the same seat (maximum of 2 passengers per seat), and where only one person, either an adult or a child, can sit. The pilot's seat, occupied by Olga, is also marked.

	O.		max 2
max 2	max 1	max 2	

What is the minimum number of flights should Olga make to bring everyone back home? One-way travel is counted as a separate flight.

Input

The first line contains two integers n and m separated by a space: the number of adults and children who set off to the lake on foot ($1 \leq n, m \leq 10^6$).

Output

Output the minimum number of flights required to return everyone home.

Examples

<i>standard input</i>	<i>standard output</i>
1 1	1
1 4	1
5 5	3

Problem H. Forbidden Set

Input file: *standard input*
 Output file: *standard output*
 Time limit: 2 seconds
 Memory limit: 512 mebibytes

A set of decimal digits is given. Find the smallest prime number that has the following property: in the decimal representation of this number, **none** of the digits belong to the given set.

For example, if the set is $\{0, 6, 3, 9\}$, then the prime 71 satisfies the requirement of the problem (except, perhaps, minimality), while the prime number 101 does not (it contains the digit 0 which is in the set).

Input

The first line of the input contains a single integer n : the number of digits in the set ($1 \leq n \leq 10$). Each of the following n lines contains a single integer d_i ($0 \leq d_i \leq 9$): the next element of the set. It is guaranteed that all d_i are pairwise distinct.

Output

If there are no primes without any digits from the given set in their decimal representation, output -1 . Otherwise, output the smallest such prime.

Examples

<i>standard input</i>	<i>standard output</i>
7 0 1 2 4 6 8 9	3
9 0 1 2 3 5 6 7 8 9	-1

Problem I. Colorful Cycles

Input file: *standard input*
 Output file: *standard output*
 Time limit: 8 seconds
 Memory limit: 1024 mebibytes

You are given a connected graph G with n vertices and m bidirectional edges. The graph does not contain loops and multiple edges. Every edge has its own color: red, yellow, or blue. Is there a simple cycle in the graph containing edges of all three colors?

Input

The first line contains an integer t , the number of test cases ($1 \leq t \leq 10^6$). The descriptions of test cases follow.

The first line of each description contains two integers n and m : the number of vertices and edges in the graph ($1 \leq n, m \leq 10^6$). The i -th of the following m lines contains three integers x_i , y_i , and z_i : they mean that the i -th edge connects vertices x_i and y_i and has color z_i . The color numbers are: 1 for red, 2 for yellow and 3 for blue. It is guaranteed that the graph is connected and that it contains no loops and no multiple edges.

The total number of vertices in all test cases is no more than 10^6 , the total number of edges in all test cases is no more than 10^6 .

Output

Print a single line for each test cases. If there is a cycle containing edges of all three colors, print “Yes”. Otherwise, print “No”. Each letter can be uppercase or lowercase.

Example

<i>standard input</i>	<i>standard output</i>
2	Yes
3 3	No
1 2 3	
2 3 1	
1 3 2	
5 6	
1 2 1	
2 3 1	
1 3 2	
3 4 3	
3 5 3	
4 5 3	

Problem J. Range Sets

Input file: *standard input*
 Output file: *standard output*
 Time limit: 3 seconds
 Memory limit: 512 mebibytes

This is an interactive problem.

You have n sets of integers. The sets are numbered from 1 to n . Initially, all sets are empty. Your task is to perform q queries of the following three types.

1. “+ ℓ r x ”. Add x to all sets with numbers from ℓ to r inclusive ($1 \leq \ell \leq r \leq n$).
2. “- ℓ r x ”. Remove x from all sets with numbers from ℓ to r inclusive ($1 \leq \ell \leq r \leq n$).
3. “? k ”. Print the size of set number k ($1 \leq k \leq n$).

The sets behave like regular sets: if we add an element that is already present, or remove an element that is already missing, nothing happens.

Input

The first line contains two integers n and q : the numbers of sets and queries correspondingly ($1 \leq n \leq 10^9$; $0 \leq q \leq 10^5$). The next q lines contain queries in the format described above ($1 \leq x \leq q$ in the queries).

Output

For each query of the third type, your program should print the answer on a separate line. It is guaranteed that there are at most 10 000 such queries. After outputting each answer, don't forget to print the newline character and flush the correct output buffer, or the outcome will be “**Idleness Limit Exceeded**”. To flush the buffer, one can call, for example, `fflush (stdout)` in C or C++, `System.out.flush ()` in Java, `flush (output)` in Pascal or `sys.stdout.flush ()` in Python.

Example

<i>standard input</i>	<i>standard output</i>
736 10	0
? 1	1
+ 1 5 1	2
+ 2 600 2	2
? 1	2
? 2	1
+ 1 6 2	
? 1	
? 2	
- 1 6 2	
? 4	

Problem K. Integer Half-Sum

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Consider a board with integers written on it. Initially, each integer from ℓ to r , inclusive, is written on the board exactly once. In one step, we can choose two numbers a and b on the board such that their half-sum $\frac{a+b}{2}$ is an **integer**, erase the two chosen numbers and write their half-sum on the board instead.

After zero or more steps, can we obtain a board with a single number on it? If yes, what is the maximum possible number that can be the single number left on the board?

Input

The first line contains two integers ℓ and r : the minimum and maximum numbers that are on the board initially ($1 \leq \ell \leq r \leq 100$).

Output

Print the maximum possible number that can be obtained as a single number on the board. If obtaining a single number on the board is impossible, print -1 .

Example

<i>standard input</i>	<i>standard output</i>	<i>explanation</i>
2 4	3	$\underline{2}, 3, \underline{4} \rightarrow \underline{3}, \underline{3} \rightarrow 3$

Problem L. esreveR Order

Input file: *standard input*
 Output file: *standard output*
 Time limit: 2 seconds
 Memory limit: 512 mebibytes

This is a run-twice problem.

You have an array of $n \leq 1000$ unsigned 64-bit integers. You want to quickly transmit it to another computer. To do this, you send numbers through the Internet in parallel and then reassemble the array.

However, an unexpected problem has arisen. As is known, the “network” byte order in a multi-byte number differs from the order used in modern computers. Specifically, in a modern computer, bytes are written from least significant to most significant (little-endian), and in network byte order, bytes are written from most significant to least significant (big-endian). During the conversion, each number is written as a sequence of 8 bytes, and the bytes are written in reverse order. And in some cases, due to server failures, the reverse conversion was not performed...

So, you send an array, and then receive another array where each element can arrive either in the usual little-endian order or in the network big-endian order. The order of the array elements is preserved. When transmitting, you can use no more than 1024 64-bit integers (in other words, no more than 8 **kibibytes**). Your task is to restore the original array after possible changes.

Input

If you need to transmit an array, the first line contains the word “**encode**”, the second line contains an integer n ($1 \leq n \leq 1000$), and the third line contains n integers in the range from 0 to $2^{64} - 1$.

If you need to receive an array, the first line contains the word “**decode**”, the second line contains an integer k ($k \leq 1024$), and the third line contains k integers in the range from 0 to $2^{64} - 1$. It is guaranteed that the numbers are in the same order as they were transmitted, and that each number is either transmitted unchanged or has its byte order reversed (according to the problem statement).

Output

In the case of transmitting an array, output one integer $k \leq 1024$ on the first line: the number of integers to transmit. On the second line, output k integers in the range from 0 to $2^{64} - 1$.

In the case of receiving an array, output n integers on a single line: the original array.

Examples

<i>standard input</i>	<i>standard output</i>
encode 3 15 10 2023	6 15 15 10 10 2023 2023
decode 6 15 15 10 720575940379279360 16647274547598327808 2023	15 10 2023

Note

In the lower example, in the case of receiving an array, all six numbers will be given on a single line. An additional line break is added for readability.

On each test, your program will be run twice: first for transmitting the array, and then for receiving it. The output of the first run with possible changes will be the input of the second run. Your solution passes a test if the original array is restored correctly.

Problem M. Good Splits

Input file: *standard input*
 Output file: *standard output*
 Time limit: 2 seconds
 Memory limit: 512 mebibytes

There are $2n$ distinct points on the real line, numbered from 1 to $2n$. A way to split them into n pairs $(a_1, b_1), \dots, (a_n, b_n)$ is *good* if the points in each pair can be connected by a curve such that the n curves don't intersect each other and don't intersect the real line. How many good ways to split are there? As this number may be large, output it modulo p , the given prime number.

Two ways to split are considered the same if we can reorder the pairs and reorder points in each pair so that the first way transforms into the second.

Input

The only line contains two integers N and p : the maximum number of points and the prime modulo ($1 \leq N \leq 200$; $10^8 < p < 10^9$).

Output

Print N lines: the answer to the problem for $n = 1, 2, \dots, N$. Print all answers modulo p .

Example

<i>standard input</i>	<i>standard output</i>
5 998244353	1 3 14 84 592

Problem N. Shoes

Input file: *standard input*
 Output file: *standard output*
 Time limit: 4 seconds
 Memory limit: 1024 mebibytes

Planning his vacation, Nikita decided to make it worthwhile and buy himself a new pair of shoes. To do this, he studied the assortment of stores located on the same street as his hotel, and selected n pairs of shoes to try on. Nikita knows that each pair of shoes requires k seconds to try on, and plans to allocate T seconds each day for visiting the stores. The street is a coordinate line, with a movement speed of one unit per second, and the hotel is located at the origin, where each visit to the stores must begin and end. Find the minimum number of vacation days that Nikita will need to try on all the shoes he is interested in.

Input

The first line contains three integers n , k , and T ($1 \leq n \leq 10^4$; $1 \leq k \leq T \leq 10^{18}$): the number of pairs of shoes, the time required for trying on, and the amount of daily free time that can be spent on visiting the stores. The second line contains n integers: the coordinates of the pairs of shoes planned for trying on. Note that these coordinates are not necessarily distinct: some pairs of shoes may be in the same store, and a store may be located within the hotel premises. However, it is guaranteed that each pair of shoes can be tried on in one evening.

Output

Output a single integer: the answer to the problem.

Example

<i>standard input</i>	<i>standard output</i>
5 100 400 -2 -1 -150 99 98	3