A. Sum Game

Limits: 2 sec., 512 MiB

Ana and Bob play the following game.

First, Bob chooses a sequence of n positive integers $a_1, a_2, \ldots, a_n$. Then Ana is allowed to perform the following operations.

- She may erase any of the numbers from the sequence, but she is not allowed to erase all of them (at least one number must remain).
- For each remaining number, she may put either a `+` or a `-` sign in front of it.

After that, Ana computes the sum of the remaining numbers. She wins the game if she can obtain a sum that is divisible by m. Otherwise, Bob wins.

Given n and m, determine which player has a winning strategy.

Input

The first line contains a single integer t – the number of test cases.

Each of the next t lines contains two integers n and m.

Output

For each test case, output a single line.

- `Ana` – if Ana has a winning strategy,
- `Bob` – if Bob has a winning strategy.

Constraints

$1 \leq t \leq 10^5$,

$1 \leq n, m \leq 10^{18}$.

Samples

Input (*stdin*)	Output (*stdout*)
3 1 2 2 3 3 8	Bob Ana Bob

Notes

In the first test case, Bob can choose $a_1 = 3$.

In the second test case, no matter how Bob chooses a_1, a_2, Ana has a winning strategy.

B. AND Reconstruction

Limits: 2 sec., 512 MiB

Andriana had a beautiful cyclic array $a = (a_0, a_1, \ldots, a_{n-1})$ of n integers, where $0 \le a_i < 2^b$ for all $0 \le i < n$, but unfortunately she lost it! However, for us, this is quite fortunate, as without these forgetful protagonists, the world would have many fewer reconstruction problems.

Meanwhile, Andriana's friend Bob has some useful information: he kept a notebook where he recorded bitwise AND values of consecutive segments from Andriana's array. Bob was quite bored one day, so he wrote down, for every position i and some fixed length k ($k \le n$), the value

$$x_i = a_i \text{ AND } a_{i+1} \text{ AND } \ldots \text{ AND } a_{i+k-1},$$

(where indices are taken modulo n, since the array is cyclic). It is unknown whether Bob acquired these ANDs through legal means.

Andriana wants to know what the largest value of k is such that Bob's notebook would contain enough information to uniquely reconstruct her treasured original array?

Input

The first line of the input contains two integers n and b, where n is the length of Andriana's lost array and b is the parameter defining the constraint $0 \le a_i < 2^b$.

The second line contains n integers a_i – the elements of the cyclic array.

Output

Print a single integer – the largest value of k such that the array a can be uniquely reconstructed from the array x of segment ANDs.

Constraints

$3 \le n \le 2 \cdot 10^5$,
$1 \le b \le 30$,
$0 \le a_i < 2^b$.

Samples

Input (*stdin*)	Output (*stdout*)
`6 3` `7 3 7 7 3 7`	`2`

Notes

In the sample case, $n = 6, b = 3$. The array $a = [7, 3, 7, 7, 3, 7]$ has binary representation

$$[111_2, 011_2, 111_2, 111_2, 011_2, 111_2],$$

using three bits for each element.

For $k = 2$, the array x would be $[3, 3, 7, 3, 3, 7]$ (where $x_i = a_i \text{ AND } a_{(i+1) \bmod 6}$). From the array x and the value of b, we can uniquely reconstruct a.

For $k = 3$, the array x would be $[3, 3, 3, 3, 3, 3]$, which does not uniquely determine a (for example, $[3, 3, 7, 3, 7, 7]$ would give the same x).

Therefore, for this sample, the answer is $k = 2$.

C. XOR-Excluding Sets

Limits: 3 sec., 512 MiB

We say that a set S of non-negative integers is *XOR-excluding* if the XOR of all elements in S is not in the set S itself.

Given an initially empty set A, we add n distinct positive elements a_i to it one by one. After each addition, find the number of XOR-excluding subsets of A.

Since the answer can be very large, output it modulo $10^9 + 7$.

Input

The first line of the input contains a single integer n – the number of elements to add.

The following n lines each contain a single integer a_i – the i-th element to add to the set.

Output

Print n lines. On the i-th line, print the number of XOR-excluding subsets of A after adding the first i elements, modulo $10^9 + 7$.

Constraints

$1 \le n \le 2 \cdot 10^5$,
$1 \le a_i \le 2^{60} - 1$,
all a_i are distinct.

Samples

Input (*stdin*)	Output (*stdout*)
5 1 2 3 4 5	1 2 5 11 22

Notes

After adding 1, the only XOR-excluding subset is $\{\}$ (the empty set has XOR equal to 0).

After adding 2, the XOR-excluding subsets are $\{\}$ and $\{1, 2\}$ (which has XOR equal to 3).

After adding 3, the XOR-excluding subsets are $\{\}$, $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$ and $\{1, 2, 3\}$.

After adding 4, the XOR-excluding subsets are $\{\}$, $\{1, 2\}$, $\{1, 3\}$, $\{1, 4\}$, $\{2, 3\}$, $\{2, 4\}$, $\{3, 4\}$, $\{1, 2, 3\}$, $\{1, 2, 4\}$, $\{1, 3, 4\}$ and $\{2, 3, 4\}$. Note that $\{1, 2, 3, 4\}$ has XOR equal to 3, so it is not XOR-excluding.

D. Two Options

Limits: 4 sec., 512 MiB

You are given two integers n and m, and m triplets of integers (i, j, x), where $1 \le i < j \le n, 1 \le x \le n$.

Permutation $p = (p_1, p_2, \ldots, p_n)$ of integers $1, 2, \ldots, n$ is *good* if for **all** m given triplets (i, j, x), it holds that either $p_i = x$ or $p_j = x$.

Calculate the number of good permutations, and print the answer modulo $10^9 + 7$.

Input

The first line contains two integers n and m – the size of the permutation p and the number of triplets.

The next m lines contain three integers i, j, and x, describing the triplets.

Output

Print a single number – the number of good permutations modulo $10^9 + 7$.

Constraints

$2 \le n \le 10^6$,
$1 \le m \le 10^6$,
$1 \le i < j \le n$,
$1 \le x \le n$,
all the triplets are pairwise distinct.

Samples

Input (*stdin*)	Output (*stdout*)
4 4 1 2 1 1 3 1 2 3 2 2 3 3	2
4 7 1 2 1 1 3 1 1 4 1 1 2 2 2 3 2 2 4 2 3 4 4	2

Notes

In the first sample, $n = 4, m = 4$. Good permutations must satisfy all of the following conditions.

- $p_1 = 1$ or $p_2 = 1$.

- $p_1 = 1$ or $p_3 = 1$.
- $p_2 = 2$ or $p_3 = 2$.
- $p_2 = 3$ or $p_3 = 3$.

There are two good permutations: $(1, 2, 3)$ and $(1, 3, 2)$.
In the second sample, the good permutations are $(1, 2, 3, 4)$ and $(1, 2, 4, 3)$.

E. Phone Company

Limits: 2 sec., 512 MiB

A mobile phone company offers the following promotion: each new customer may choose exactly k other phone numbers and all calls between the customer and any of the k chosen contacts are free (in both directions).

A group of n students (numbered from 1 to n) wants to take advantage of this promotion. Each student i must choose exactly k distinct contacts from the set $\{1, 2, \ldots, n\} \setminus \{i\}$.

We say that two students i and j can talk for free if at least one of the following holds:

- i has chosen j as one of the free contacts, or
- j has chosen i as one of the free contacts.

The students want to choose their free contacts so that any student can talk to any other student for free, i.e., for every pair (i, j) with $1 \leq i < j \leq n$, students i and j can talk for free.

For a given k, your task is to:

- determine the maximum possible number n of students for which such a configuration is possible,
- construct one valid configuration of chosen contacts for this maximum n.

Input

The input consists of a single integer k.

Output

On the first line, output a single integer n – the maximum number of students for which it is possible to choose the free contacts so that any two students can talk to each other for free.

On the next n lines, output the k free contacts chosen by each student: on line $i+1$ (for $1 \leq i \leq n$), output k distinct integers $a_{i,1}, a_{i,2}, \ldots, a_{i,k}$. Each $a_{i,j}$ must satisfy $1 \leq a_{i,j} \leq n$ and $a_{i,j} \neq i$. These lines indicate that student i has chosen the students $a_{i,1}, a_{i,2}, \ldots, a_{i,k}$ as their free contacts.

If there are several valid answers, print any of them.

Constraints

$1 \leq k \leq 1000$.

Samples

Input (*stdin*)	Output (*stdout*)
2	5 2 5 3 4 1 5 1 3 2 4

Notes

In the example, $k = 2$ and the maximum number of students for which it is possible to choose the free contacts so that any two students can talk to each other for free is $n = 5$. For each pair of distinct students, at least one of them has chosen the other, so any two can talk for free.

F. Language Barrier

Limits: 5 sec., 512 MiB

It's not about the money, it's about sending a message.

In a network of n people connected by a tree structure, person 1 wants to send a message to person n. There are 10^9 different languages numbered from 1 to 10^9. Person i speaks all languages in the interval $[l_i, r_i]$.

Person 1 starts with a piece of paper and can write the initial message in any language they speak. On each turn, the person currently holding the paper can do one of the following.

- Pass the paper to a neighboring person in the tree.
- If they understand the language currently written on the paper, translate it to any other language they speak (overwriting the original).

Find the minimum number of turns for person n to receive the paper with a message written in a language they understand.

Input

The first line contains a single integer n – the number of people.

The next $n - 1$ lines each contain two integers u and v, describing an edge between people u and v in the tree.

The next n lines each contain two integers l_i and r_i – the interval describing the languages that person i speaks.

Output

Print a single integer – the minimum number of turns needed for person n to receive the message in a language they understand.

Constraints

$2 \leq n \leq 2 \cdot 10^5$
$1 \leq u, v \leq n$ for all edges,
$u \neq v$ for all edges,
$1 \leq l_i \leq r_i \leq 10^9$,
it is guaranteed that any person can send a message to any other person (possibly requiring a series of translations).

Samples

Input (*stdin*)	Output (*stdout*)
4 1 2 2 3 3 4 1 3 2 4 3 5 4 6	4

Notes

In the sample, the tree is a path.

Person 1 speaks languages 1, 2, 3. Person 2 speaks languages 2, 3, 4. Person 3 speaks languages 3, 4, 5. Person 4 speaks languages 4, 5, 6.

One optimal solution is as follows. First, person 1 writes the message in language 3. Then the people start to make turns.

1. Person 1 passes the paper to person 2.
2. Person 2 translates the message from language 3 into language 4.
3. Person 2 passes the paper to person 3.
4. Person 3 passes the paper to person 4.

Person 4 now has the message in language 4, which they understand. The minimum number of turns needed is four.

G. Intervals from Triplets

Limits: 3 sec., 512 MiB

You are given n triplets of integers (a_i, b_i, c_i), such that $a_i < b_i < c_i$.

You have to choose a pair of values from each triplet and therefore form n intervals. You need to choose pairs such that every two distinct intervals have zero intersection length.

Find the maximum possible sum of lengths of intervals, or tell that it is impossible to choose intervals satisfying the condition.

Input

The first line contains one integer n – the number of triplets.

The next n lines contain three integers a_i, b_i, and c_i – the i-th triplet.

Output

If it is impossible to form intervals that satisfy the condition, print `-1`. Otherwise, print the maximum possible sum of lengths of intervals.

Constraints

$1 \le n \le 10^6$,
$1 \le a_i < b_i < c_i \le 10^9$.

Samples

Input (*stdin*)	Output (*stdout*)
4 1 2 3 2 3 5 7 8 9 4 7 8	7
3 1 2 4 1 2 4 1 2 4	-1

Notes

In the first sample, there are $n = 4$ triplets: $(1, 2, 3)$, $(2, 3, 5)$, $(7, 8, 9)$, and $(4, 7, 8)$.

One optimal solution is: $(1, 2)$, $(2, 3)$, $(7, 9)$, $(4, 7)$. These intervals satisfy the condition that every two distinct intervals have zero intersection length. The sum of their lengths is $(2-1)+(3-2)+(9-7)+(7-4) = 1+1+2+3 = 7$.

In the second sample, it is impossible to choose the intervals satisfying the condition.

H. Sorted Pairs

Limits: 4 sec., 512 MiB

You are given an integer matrix a of size $n \times 2$. Elements in the first column of matrix a form a permutation of values $n+1, n+2, \ldots, 2n$, and elements in the second column form a permutation of values $1, 2, \ldots, n$.

You can do any number of moves. In one move you can do the following.

- Choose any positive integer k.
- Choose a sequence of k distinct numbers $(b_1, b_2, \ldots, b_k)$, such that $1 \le b_i \le 2n$ for all $1 \le i \le k$.
- Cyclically shift all chosen numbers in the matrix (number b_1 goes to the position of b_2, number b_2 goes to the position of b_3, ..., number b_k goes to the position of b_1).
- Pay k coins.

What is the minimum number of coins you have to pay to get the left element smaller than the right element in every row? Note that it is not required to minimize the number of moves.

Input

The first line contains an integer t – number of test cases.
The first line of each test case contains an integer n – the number of rows in matrix a.
The next n lines of each test case contain two integers $a_{i,1}$ and $a_{i,2}$ – elements of the matrix.

Output

For each test case, print the answer in the following format.
On the first line print the minimum number of coins.
On the second line print an integer m – the number of moves.
On the next m lines print the moves in the format $k\ b_1\ b_2\ b_3 \ldots b_k$.

Constraints

$1 \le t \le 10^6$,
$1 \le n \le 10^6$,
the sum of n over all test cases does not exceed 10^6.

Samples

Input (*stdin*)	Output (*stdout*)
2 4 5 1 6 2 7 3 8 4 3 4 2 5 1 6 3	4 1 4 1 7 2 8 4 2 2 1 6 2 4 2

Notes

In the first sample, the matrix $a = \begin{pmatrix} 5 & 1 \\ 6 & 2 \\ 7 & 3 \\ 8 & 4 \end{pmatrix}$ is given.

One of the optimal strategies for this matrix is to make one move choosing $k = 4, b = (1, 7, 2, 8)$. During this move

- 1 goes to the position of 7,
- 7 goes to the position of 2,
- 2 goes to the position of 8,
- 8 goes to the position of 1.

After the move, the matrix a becomes $\begin{pmatrix} 5 & 8 \\ 6 & 7 \\ 1 & 3 \\ 2 & 4 \end{pmatrix}$. Now, it holds that in every row, the left element is smaller than the right element.

The total number of paid coins is four.

I. Colorful Components

Limits: 2 sec., 512 MiB

You are given an undirected simple graph $G = (V, E)$ with $|V| = n$ vertices and $|E| = m$ edges. Each vertex $v \in V$ has a color c_v, represented by an integer.

A *spanning subgraph* of G is a graph $G' = (V, E')$ with $E' \subseteq E$, i.e. it uses the same set of vertices but an arbitrary subset of the edges.

A connected component of a graph is called *colorful* if no two vertices in that component have the same color. In other words, for every color γ, there is at most one vertex of color γ in the component.

A vertex is called a *singleton* in a graph if it has degree 0 (it is an isolated vertex, forming a component of size 1).

Your task is to choose a subset of edges $E' \subseteq E$ and construct a spanning subgraph $G' = (V, E')$ such that:

- Every connected component of G' is colorful.
- The number of singleton vertices in G' is as small as possible.

You must output the minimum possible number of singleton vertices and one corresponding spanning subgraph achieving this minimum.

Input

The first line contains two integers n and m – the number of vertices and edges in the graph.

The second line contains n integers $c_1, c_2, \ldots, c_n$, where c_i is the color of vertex i.

Each of the next m lines contains two integers u and v, denoting an undirected edge between vertices u and v.

Output

On the first line, print a single integer s – the minimum possible number of singleton vertices among all spanning subgraphs $G' = (V, E')$ in which every connected component is colorful.

On the second line, print an integer k – the number of edges in your chosen set E'.

Then print k lines, each containing two integers u and v – the edges of E'.

Any spanning subgraph $G' = (V, E')$ that satisfies the conditions and yields exactly s singleton vertices will be accepted.

Constraints

$1 \le n \le 5 \cdot 10^3$,
$1 \le m \le 5 \cdot 10^3$,
$1 \le c_i \le n$,
the given graph does not contain multiple edges or self-loops.

Samples

Input (*stdin*)	Output (*stdout*)
4 4 1 1 2 2 1 3 1 4 2 3 2 4	0 2 1 3 2 4

Notes

In the example, one optimal solution is to choose edges $(1, 3)$ and $(2, 4)$. Then there are two connected components: $\{1, 3\}$ and $\{2, 4\}$. Each component contains two vertices of different colors, hence both components are colorful. Every vertex has degree at least 1, so there are 0 singleton vertices, which is optimal.

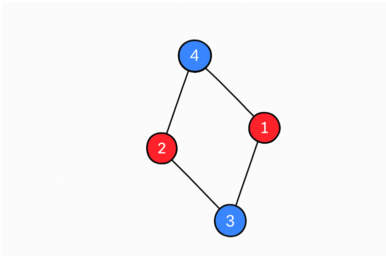


Figure 1: The input graph G in the example.

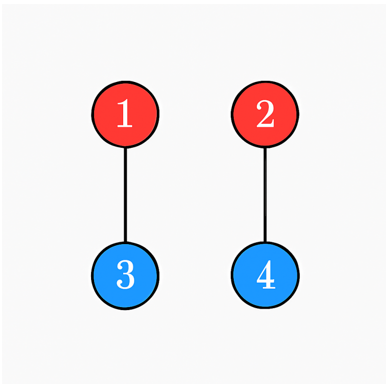


Figure 2: An optimal colorful spanning subgraph G' of G.

J. Triangle Hull

Limits: 3 sec., 512 MiB

There are n points p_i on a two-dimensional plane, with p_i located at coordinates (x_i, y_i). It is guaranteed that no two points occupy the same position, and no three points are collinear.

Find the number of ways to choose three points from the given set such that the convex hull of any subset containing these three points is a triangle.

Input

The first line contains an integer n – the number of points.

Each of the next n lines contains two integers x_i and y_i – the coordinates of p_i.

Output

Output a single integer – the number of ways to choose three points from the given set such that the convex hull of any subset containing these three points is a triangle.

Constraints

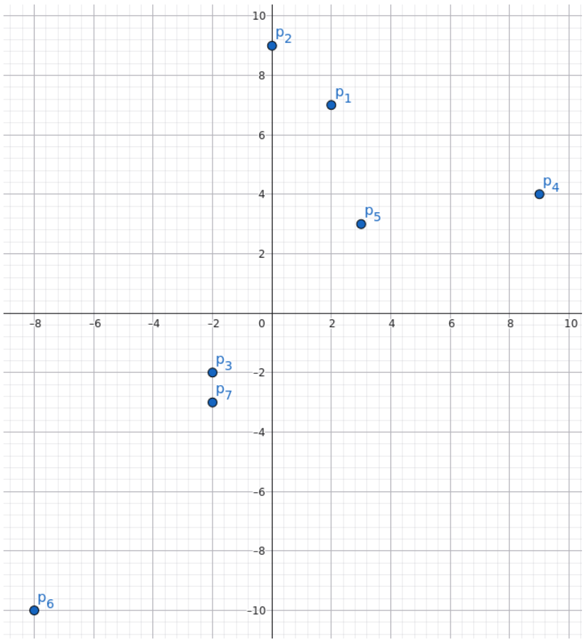
$3 \le n \le 2000$,
$|x_i|, |y_i| \le 10^9$,
no two points are at the same location,
no three points are collinear.

Samples

Input (*stdin*)	Output (*stdout*)
7 2 7 0 9 -2 -2 9 4 3 3 -8 -10 -2 -3	2
4 -1 -1 1 -1 0 1 0 0	4
7 -50 -33 50 -31 -93 98 -47 -59 16 -35 79 -25 -75 41	15

Notes

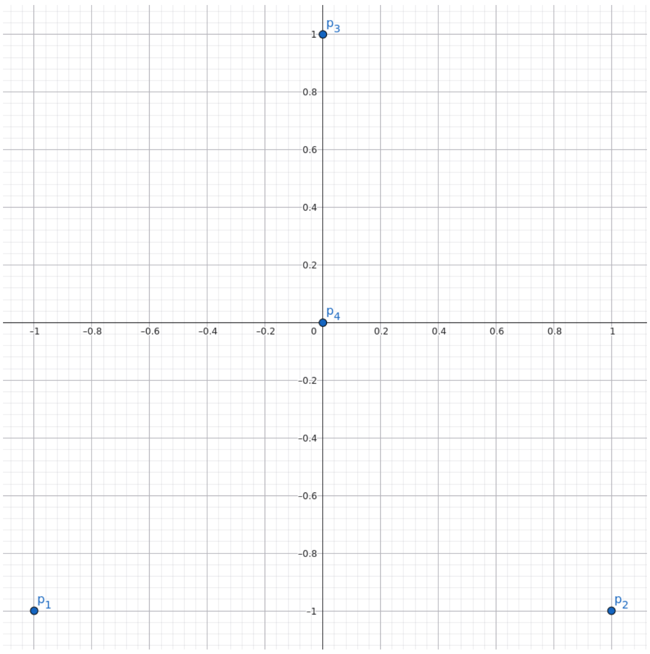
The given points in the first sample are arranged as shown in the figure below.



In the first sample, the following two ways to choose three points satisfy the condition:

- $\{p_1, p_4, p_6\}$,
- $\{p_2, p_4, p_6\}$.

The given points in the second sample are arranged as shown in the figure below.



In the second sample, the following four ways to choose three points satisfy the condition:

- $\{p_1, p_2, p_3\}$,
- $\{p_1, p_2, p_4\}$,
- $\{p_1, p_3, p_4\}$,
- $\{p_2, p_3, p_4\}$.

K. Connect the Points

Limits: 2 sec., 512 MiB

There are n pairs of points with integer coordinates on a two-dimensional plane. The i-th pair contains points $(x_1^{(i)}, y_1^{(i)})$ and $(x_2^{(i)}, y_2^{(i)})$. The $2n$ points are distinct and each of them lies in the square $[0, n] \times [0, n]$.

Determine whether it is possible to connect every pair of points with a line (not necessarily a straight line) such that the following conditions hold.

- Every drawn line stays strictly inside or on the boundary of the square $[0, n] \times [0, n]$.
- No two lines intersect.

Input

The first line contains an integer n – the number of pairs.

Each of the next n lines contains four integers $x_1^{(i)}, y_1^{(i)}, x_2^{(i)}, y_2^{(i)}$ – the coordinates of the two points of the i-th pair.

Output

Print `YES`, if it is possible to connect all pairs without intersections, or `NO` otherwise.

Constraints

$1 \le n \le 7$,
$0 \le x_j^{(i)}, y_j^{(i)} \le n$ for all $1 \le i \le n, 1 \le j \le 2$,
all $2n$ points are distinct.

Samples

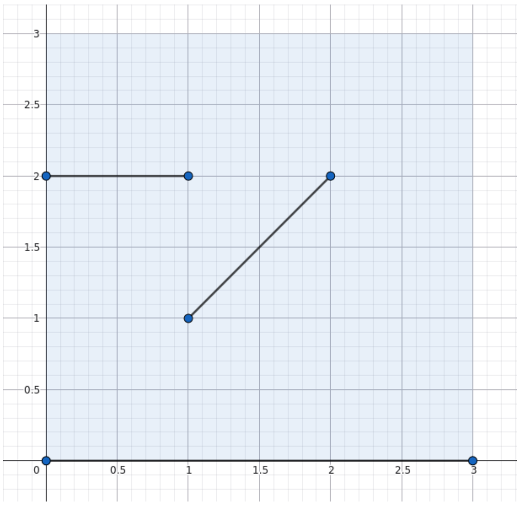
Input (*stdin*)	Output (*stdout*)

`3`
`0 0 3 0`
`1 1 2 2`
`0 2 1 2`
	`YES`

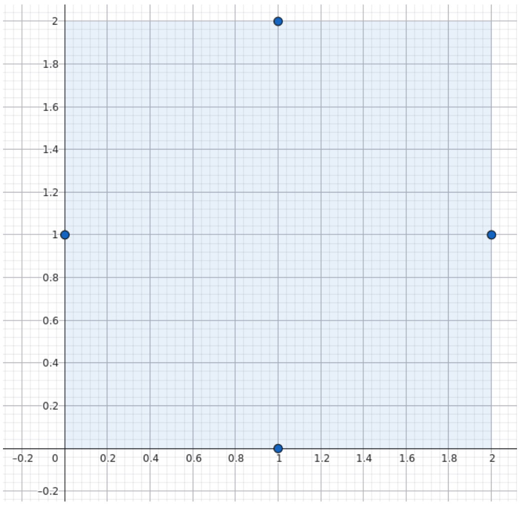
`2`
`0 1 2 1`
`1 0 1 2`
	`NO`

`2`
`0 0 2 2`
`2 0 1 1`
	`YES`

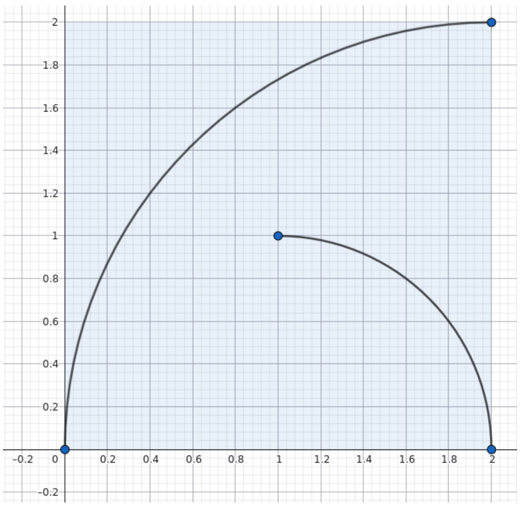
Notes



In the first sample, it is possible to connect the points as shown in the picture above.



In the second sample, it is impossible to connect the points such that the conditions are satisfied.



In the third sample, it is possible to connect the points as shown in the picture above.

L. Neo-Nim

Limits: 2 sec., 512 MiB

There are n piles of stones. The i-th pile initially contains a_i stones. Additionally, an integer k is given.

Ana and Bob play a game with these stones, alternating turns with Ana going first. The moves are defined as follows.

- **Ana** chooses an integer x such that $2 \le x \le k$ and a pile containing at least x stones, then removes exactly x stones from it.
- **Bob** chooses a pile containing at least one stone and removes exactly one stone from it.

The player who cannot make a move loses. Determine the winner assuming both players play optimally.

Input

The first line contains an integer t – the number of test cases.

The first line of each test case contains two integers n and k – the number of piles and the maximum limit for Ana's move.

The second line of each test case contains n integers a_i – the number of stones in each pile.

Output

For each test case, output `Ana` if Ana wins, and `Bob` if Bob wins.

Constraints

$1 \le n \le 10^5$,
$2 \le k \le 10^5$,
$1 \le a_i \le 10^5$,
the sum of n over all test cases does not exceed $3 \cdot 10^5$.

Samples

Input (*stdin*)	Output (*stdout*)
3 3 2 2 5 8 4 3 2 2 4 5 1 5 2	Bob Bob Ana

Notes

In the first sample, $n = 3, k = 2, a = (2, 5, 8)$.
One of the possible game scenarios could be as follows.

- Ana removes two stones from the third pile. After this move a becomes $(2, 5, 6)$.

- Bob removes one stone from the first pile. After this move a becomes $(1, 5, 6)$.
- Ana removes two stones from the second pile. After this move a becomes $(1, 3, 6)$.
- Bob removes one stone from the third pile. After this move a becomes $(1, 3, 5)$.
- Ana removes two stones from the third pile. After this move a becomes $(1, 3, 3)$.
- Bob removes one stone from the third pile. After this move a becomes $(1, 3, 2)$.
- Ana removes two stones from the second pile. After this move a becomes $(1, 1, 2)$.
- Bob removes one stone from the third pile. After this move a becomes $(1, 1, 1)$.
- It is Ana's turn now, but she cannot make a move. Ana loses and Bob wins.

In the first sample, Bob wins.
In the second sample, Bob wins.
In the third sample, Ana wins.