Problem A. Reversing

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

Consider a rectangular grid of size $N \times M$. Each cell is colored black or white.

If you touch a cell C of the grid, you change the color of all the cells that belong to same-colored connected component of C, including C itself. For connected components, two cells are neighbors if they share a side.

You know the current state of the grid, but you may have touched some cells an arbitrary number of times. Calculate the number of possible initial states of the grid. As the answer may be very large, calculate it modulo 1 000 000 007.

Input

The first line contains two integers N and M, the dimensions of the grid ($1 \le N, M \le 2000$).

Each of the next N lines describes one row of the grid. Each of these lines contains M characters denoting the colors of cells in the row. Each character is either "`B`" for black or "`W`" for white.

Output

Print the number of possible initial states of the grid modulo 1 000 000 007.

Example

standard input	*standard output*
2 2 WW WB	2

Problem B. Lawyers

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

There are N lawyers. Each lawyer has been charged with committing a fraudulent offense. These N lawyers try to defend each other and make sure they are acquitted.

Lawyer A can defend lawyer B if and only if lawyer B trusts lawyer A, and there are M such pairs (A, B). Note that, if lawyer B trusts lawyer A, it does not imply that lawyer A trusts lawyer B.

Each lawyer is very hard-working, so one lawyer can defend any number of others.

Each lawyer is very talented, so anyone who receives at least one defense is unconditionally acquitted. With one exception: if lawyer A defends lawyer B and lawyer B defends lawyer A, it seems very suspicious, and both are found guilty.

Determine whether it is possible or not for all lawyers to be acquitted together.

Input

The first line contains two integers N and M, the number of lawyers and the number of trust relationships ($1 \le N, M \le 200\,000$).

The next M lines describe trust relations. The i-th of these M lines contains two different integers A_i and B_i, which means lawyer B_i trusts lawyer A_i, and so lawyer A_i can defend lawyer B_i. There are no such i and j ($1 \le i, j \le M$, $i \ne j$) that $A_i = B_i$ and $A_j = B_j$.

Output

Print "`YES`" (without quotes) if it is possible for all lawyers to be acquitted together. Print "`NO`" (without quotes) otherwise.

Examples

standard input	*standard output*

3 3
1 2
2 3
3 1
	YES

4 6
1 2
1 3
1 4
2 3
2 4
3 4
	NO

4 4
1 2
2 1
3 4
4 3
	NO

Problem C. One, Two, Three

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

You are given a sequence of length N: $A_0, A_1, \ldots, A_{N-1}$. It consists of only three kinds of integers: $1, 2, 3$.

A tuple of indices (i, j, k) is *good* if $0 \le i < j < k < N$ and it satisfies one of the two following conditions: either $A_i = 1$, $A_j = 2$, $A_k = 3$ or $A_i = 3$, $A_j = 2$, $A_k = 1$.

Your goal is find disjoint good tuples, as many of them as possible. A group of tuples is disjoint if no index is present in more than one tuple.

Find the maximum number of disjoint good tuples and print each tuple.

Input

The first line contains an integer N, the length of the given sequence ($1 \le N \le 600\,000$).

The next line contains N integers: $A_0, A_1, \ldots, A_{N-1}$ ($1 \le A_i \le 3$).

Output

On the first line, print an integer M, the maximum number of disjoint good tuples.

On the next M lines, print the tuples themselves. Each of these lines must contains three integers i, j, k ($0 \le i < j < k < N$) that describes a good tuple. All the printed tuples must be disjoint. If there are several solutions, print any one of them.

Examples

standard input	*standard output*

6
3 1 2 2 3 1
	2
	1 2 4
	0 3 5

6 2 1 3 1 3 2	0

Problem D. Lonely King

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

You are given a rooted tree with N vertices. Vertex 1 is the root, and each of the other $N-1$ vertices has exactly one incoming edge. There are C_i people living in i-th vertex.

Initially, all edges are blue. You can change a "blue path" into a "red edge". Formally, when there are k blue edges, (a_1, a_2), $(a_2, a_3), \ldots, (a_k, a_{k+1})$, you can replace them with one red edge, (a_1, a_{k+1}). You can execute this operation any number of times.

Because of the COVID-19, your purpose is to prevent contacts between people, so you want to minimize the total number of contacts.

The total number of contacts is the number of pairs of people (A, B) such that A and B live in different vertices and A can visit B via edges (of any color). Note that the edges are directed.

Find the minimum total number of contacts that can be achieved after some (possibly zero) operations on the tree.

Input

The first line contains an integer N, the number of vertices ($1 \le N \le 200\,000$).

The next line contains $N-1$ integers, $P_2, P_3, \ldots, P_N$ ($1 \le P_i \le N$). It means that vertex i has one incoming edge from vertex P_i. These numbers describe a rooted tree with vertex 1 as the root. Keep in mind that the edges are directed.

The next line contains N integers, $C_1, C_2, \ldots, C_N$, which denote the number of people in each vertex ($1 \le C_i \le 10^6$).

Output

Print one integer, the minimum total number of contacts.

Example

standard input	*standard output*

4
1 1 2
2 1 3 2
	10

Problem E. Treasure Box

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

You are playing a VR game about adventures in an ancient world. You were on a journey to find the legendary treasure, and after breaking through several gates, you found a box that was supposed to contain the treasure. But the box had a lock on it with the following message: "To open the box, change the letters under the box well, so that it reads the same forward and backward."

After looking under the box, you found a string consisting of N characters in a row. The i-th character is located at position $(i, 0)$, so distances between adjacent characters are 1.

It seems that the box can be opened by replacing the characters under the box and making it a palindrome. To do that, you start at some position, and you can repeatedly move to another position and replace the character at that position by any other character, until the string becomes a palindrome.

Since your HP is limited, you want to gain the treasure using minimum HP. Each position requires a different amount of HP to replace the respective character. Also, it consumes C HP to move a unit distance. That is, if you were at position $(i, 0)$, and you want to move to position $(j, 0)$ to replace the j-th character, the movement will decrease your HP by $C \cdot |j - i|$.

For each integer i such that $1 \le i \le N$, find the minimum HP consumed to obtain the treasure if you start at position $(i, 0)$.

Input

The first line contains an integer T, the number of test cases ($1 \le T \le 100\,000$). The test cases follow.

The first line of each test case contains two integers: N, the number of characters under the box ($1 \le N \le 1\,000\,000$), and C, the amount of HP consumed when moving a unit distance ($1 \le C \le 10^9$).

The second line consists of N characters. It represents the string under the box. Each letter is an uppercase English letter.

The third line contains N integers. The i-th integer represents the HP consumed to replace the i-th character. Each of these integers is between 1 and 10^9.

The sum of N over all test cases does not exceed $1\,000\,000$.

Output

For each test case, print N integers on a single line. The i-th integer must be the minimum HP consumed when starting at position $(i, 0)$.

Example

standard input	*standard output*
2 5 1 ABCDE 7 1 4 5 1 5 1 ABCDA 7 1 4 5 1	6 5 6 6 5 2 1 2 3 4

Note

For the first test case, when the starting position is $(1, 0)$, one of the optimal ways is to first move to position $(2, 0)$ and change "B" to "D", then move to position $(5, 0)$ and change "E" to "A".

Problem F. Beautiful Sequence

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

There is a sequence consisting of N integers. We want to rearrange the integers to make the most beautiful sequence possible. A sequence is more beautiful when there are more members which are not less than their neighbors. The *beauty* of a sequence is the number of such members.

Write a program that will rearrange a given sequence to make it the most beautiful possible.

For example, if $N = 6$ and the sequence is $1, 1, 2, 3, 3, 4$, the beauty of the given sequence is 3. However, if we rearrange the sequence to become $2, 1, 3, 3, 1, 4$, then the beauty of the rearranged sequence is 4, which is the maximum possible.

Input

The first line contains an integer T, the number of test cases ($1 \le T \le 2222$). The test cases follow.

The first line of each test case contains an integer N, the number of elements ($1 \le N \le 300\,000$).

The next line contains the elements of the sequence. Each element is an integer between 1 and 10^9, inclusive.

The sum of N over all test cases does not exceed $5\,000\,000$.

Output

For each test case, print one line containing an integer: the highest beauty possible after rearrangement.

Example

standard input	*standard output*

2
6
1 1 2 3 3 4
5
1 2 2 3 3
	4
	4

Problem G. Make Everything White

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

Consider a rectangular grid of size $N \times M$. Each cell is colored black or white.

For each cell, you have to execute exactly one of the three operations given below.

1. Do not change anything.
2. Change the color of all neighbor cells (black to white, white to black).
3. Change the color of all neighbor cells and itself.

Two cells are neighbors if they share a side.

Your goal is make all cells white. Find a way to achieve it, or determine that it is impossible.

Input

The first line contains two integers N and M, the dimensions of the grid ($1 \le N, M \le 2000$).

Each of the next N lines describes one row of the grid. Each of these lines contains M characters denoting the colors of cells in the row. Each character is either "`B`" for black or "`W`" for white.

Output

Print "`-1`" on the first line if it is impossible to make all cells white.

Otherwise, print "`1`" on the first line, followed by N more lines. Each of these lines must contain M characters which describe the operations you chose for the respective row of the grid. Each of the characters must be either "`1`", "`2`", or "`3`", corresponding to the three operations in the statement. If there are several solutions, print any one of them.

Examples

standard input	*standard output*
2 3 WBW BWB	1 111 121
1 1 B	1 3

Problem H. Optimal Quadratic Function

Input file:	*standard input*
Output file:	*standard output*
Time limit:	10 seconds
Memory limit:	1024 mebibytes

Two variables x and y are dependent to each other with the relation $y = f(x)$ where f is a quadratic function: $f(x) = ax^2 + bx + c$ with some real numbers a, b, and c. However, the function f is unknown and you want to figure out its best estimation.

For that purpose, you have obtained N observed y-values $y_1, y_2, \ldots, y_N$ for x-values $x_1, x_2, \ldots, x_N$, respectively, by experiments. The observed values $y_1, y_2, \ldots, y_N$ contain some errors from several sources, so it is unlikely that all of them are exact function values for a certain quadratic function. Therefore, you need to find an optimal estimation of the function f that minimizes the error.

For any quadratic function f, the error of a data pair (x_i, y_i) is defined to be $(y_i - f(x_i))^2$, and the error of f is defined to be the maximum of these errors over all the N data pairs. Write a program that, given the N observed data pairs, finds out an optimal estimation of function f that minimizes the error and prints out the error value.

Input

The first line contains an integer T, the number of test cases ($1 \le T \le 100\,000$). The test cases follow.

The first line of each test case contains an integer N, the number of observed data pairs ($1 \le N \le 100\,000$).

Each of the next N lines contains two integers x_i and y_i, the i-th data pair ($-10^6 \le x_i, y_i \le 10^6$).

The sum of N over all test cases does not exceed $200\,000$.

Output

For each test case, print a line with a real number: the minimum possible error value.

The answer will be considered correct if its absolute or relative error is within 10^{-6}.

Example

standard input	*standard output*

1
4
0 0
1 3
2 9
3 0
	5.062500000000

Problem I. Visiting Friend

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 mebibytes

The village where Sunghyeon lives consists of N intersections and M two-way roads. The intersections are numbered by integers from 1 to N.

Each two-way road connects two different intersections, there is a path along the roads between any two intersections. Also, for any two intersections, there is at most one road directly connecting them.

On Sunday, Sunghyeon left her house and went to play at the house of her friend Changki, who lives in the same village. Since Sunghyeon has just moved in, she doesn't know anything about the village's road network, and she can only distinguish her house and Changki's house: all other houses look the same to her. After wandering for a long time, she finally arrived at Changki's house.

Sunghyeon remembers that, after departure, she did not visit the intersection where her house was located anymore, and she immediately entered Changki's house when she reached the intersection where Changki's house was located.

Sunghyeon wondered how many intersections she might have visited when she went to play at Changki's house. In other words, out of N intersections, she wants to count such intersections V that there exists a possible path for her which visited V.

Curious Sunghyeon went a step further here, and she wondered how many different intersections she might have visited if her house was at intersection A and Changki's house was at intersection B. She wants the answer for many different pairs (A, B).

Write a program to help Sunghyeon and Changki.

Input

The first line contains an integer T, the number of test cases ($1 \le T \le 1000$). The test cases follow.

The first line of each test case contains two integers N and M, the number of intersections and the number of roads ($2 \le N \le 200\,000$, $1 \le M \le 500\,000$).

Each of the next M lines contains two integers U_i and V_i which mean there is a two-way road between these intersections ($1 \le U_i, V_i \le N$, $U_i \ne V_i$). There is a path along the roads between any two intersections. Also, for any two intersections, there is at most one road directly connecting them.

The next line contains an integer Q, the number of questions ($1 \le Q \le 500\,000$).

Each of the next Q lines contains two integers A_j and B_j which mean you have to find the number of different intersections Sunghyeon might have visited if her house was at intersection A_i and Changki's house was at intersection B_i ($1 \le A_j, B_j \le N$, $A_j \ne B_j$).

The sum of N over all test cases does not exceed $200\,000$. The sum of M over all test cases does not exceed $500\,000$. The sum of Q over all test cases does not exceed $500\,000$.

Output

For each test case, print Q lines. The i-th of these Q lines must contain the number of intersections Sunghyeon might have visited if her house was at intersection A_i and Changki's house was at intersection B_i.

Example

standard input	*standard output*

1
5 5
1 2
1 3
2 4
4 5
2 5
5
1 2
1 4
2 3
2 5
3 5
	2
	4
	3
	3
	5

Problem J. Cooperation Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

Athletic contests in schools sometimes get too heated. To promote cooperation, the teachers have devised a game where cooperation leads to better scores.

This cooperation game starts with N students standing in a line. To get a score, two students from the same class will get out of the line. If they were i-th and j-th students along the line, just before they went out, they add $|i - j|$ to the total score. The empty places of the two students that just came out will be compacted. The game stops when there are no more pairs of students from the same class.

For example, consider six students initially standing in a line:

$$1, 2, 3, 3, 2, 1$$

The numbers are class numbers of the students. If the students come out from the line in the order of class number 1, then class number 2, then class number 3, the total score is $5 + 3 + 1 = 9$. However, in the same initial situation, if the students come out from the line in the order of 3, 2, 1, then the total score is $1 + 1 + 1 = 3$.

Given the class numbers in the initial line, write a program to calculate the maximum possible score.

Input

The first line contains an integer T, the number of test cases ($1 \leq T \leq 48$). The test cases follow.

The first line of each test case contains a single integer N, the number of students ($1 \leq N \leq 300\,000$).

In the next line, the sequence of class numbers are given. The class numbers are integers from 1 to N.

The sum of N over all test cases does not exceed $7\,000\,000$.

Output

For each test case, print a line with a single integer: the maximum possible score.

Example

standard input	*standard output*

2
7
1 2 1 1 2 1 2
12
1 2 3 1 2 3 1 2 3 1 2 3
	10
	30

Problem K. Connect the Dots

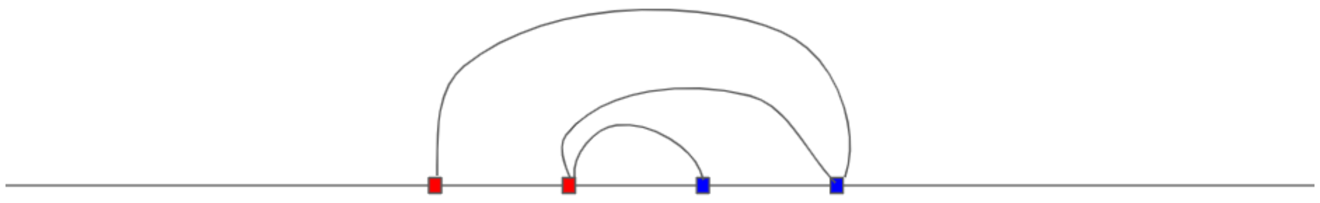
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 mebibytes

Consider N different points on the Ox axis, numbered $1, 2, \ldots, N$ from left to right. Each point has a color: the color of point i is A_i.

You want to draw several curves, each curve connecting two points. However, there are the following restrictions.

- Two points of the same color cannot be connected.
- Each curve connecting the points must be above the x-axis. In other words, each interior point of each curve has $y > 0$. (Endpoints have $y = 0$.)
- Two different curves cannot have a common interior point. (It is possible to share endpoints.)

For example, if there are 4 points as shown below, points 1 and 2 are red, and points 3 and 4 are blue, you can draw a total of 3 curves: between points 1 and 4, 2 and 3, 2 and 4.



Drawing 4 curves would violate at least one of the three restrictions above, so 3 is the maximum in this case.

Given the color of each point, find a way to draw as many curves connecting two points as possible without violating any restrictions, and print which two points each curve connects.

Input

The first line contains an integer T, the number of test cases ($1 \le T \le 101$). The test cases follow.

The first line of each test case has the number of points N and the number of colors M ($2 \le N \le 200\,000$, $2 \le M \le N$).

The next line contains N integers $A_1, A_2, \ldots, A_N$ ($1 \le A_i \le M$).

The sum of N over all test cases does not exceed $200\,000$.

Output

For each test case, start with a line containing an integer K: the maximum number of curves connecting two points.

In each of the next K lines, print the indices of the two points connected by a curve. The curves must satisfy all the restrictions above. If there are several possible answers, print any one of them.

Example

standard input	*standard output*

3
4 2
1 1 2 2
4 2
1 2 1 2
3 3
1 2 3
	3
	2 3
	2 4
	4 1
	4
	1 2
	2 3
	3 4
	4 1
	3
	3 1
	1 2
	2 3