A: Video Transmission

Time limit: 1.0 sec

Memory limit: 256 MB

You have n videos that need to be sent one by one through a communication link. The size of the i-th video is b_i, and it needs to be sent at a constant rate within t_i seconds.

You cannot send two videos simultaneously, so you need to determine the order in which these videos are sent. Regardless of the sending order, you will start sending the videos at time 0, and there should be no delay between the end of sending one video and the start of sending the next one: in other words, regardless of the sending order, all videos should finish sending at time $\sum t_i$.

As you are an ordinary user, the link can only provide you with the bandwidth with an average rate of r, so there are the following restrictions on the rate at which you send videos:

- For any positive real number T, the total size of the videos you send between time 0 and time T must be less than or equal to $r \times T$. Note that this constraint only applies to time intervals starting from 0, not from any other values.

You want to send these videos as quickly as possible. Given the parameters of all the videos to be sent, please find a sending order that satisfies the above rate limit, or report that no such sending order exists.

Input Format:

The first line of input contains two positive integers separated by a space, n and r $(1 \leq n \leq 10^6, 1 \leq r \leq 10^9)$, representing the total number of videos you need to send and the average bandwidth, respectively.

The second line of input contains n positive integers separated by spaces. The i-th integer b_i $(1 \leq b_i \leq 10^9, 1 \leq i \leq n)$ represents the size of the i-th video.

The third line of input contains n positive integers separated by spaces. The i-th integer t_i $(1 \leq t_i \leq 10^9, 1 \leq i \leq n)$ represents the time in seconds within which the i-th video needs to be sent at a constant rate.

Output Format:

If there is no sending order that satisfies the restriction, output a single line containing the string `No`.

If there is a sending order that satisfies the restriction, output two lines. The first line contains the string `Yes`. The second line contains n positive integers separated by spaces, where the i-th integer s_i represents

the i-th video sent (i.e., the videos are sent from left to right according to the order of the output numbers). Your schedule must satisfy the above rate limit, and each video must be sent exactly once.

The outputs, `Yes` or `No`, are not case-sensitive.

Examples

Input	Output
3 2500 2000 6000 2000 1 2 1	Yes 3 1 2
3 2000 2000 6000 2000 1 2 1	No

Hints

Since there is a large number of inputs and outputs. You'd better choose a fast way to input and output data. For example, if you use `C/C++`, you may use `scanf` and `printf`. Otherwise, your answer may be time out.

In the first example, the orders that meet the requirements are 1 3 2 and 3 1 2.

In the second example, whatever the order you use for sending the videos, when $T = 4$, the total size of the sent is 10000, which is over the limit of the link (at this moment, the limit is $r \times T = 2000 \times 4 = 8000$).

B: Convolution

Time limit: 1.0 sec

Memory limit: 256 MB

Given the values of an integer sequence $a_1, a_2, \cdots, a_n$, find the value of $\sum_{i=1}^{n} \sum_{j=1}^{n} 2^{a_i * a_j}$. Since the answer may be large, you only need to output the value of the answer modulo 998244353.

Input Format

The first line contains a single positive integer n.

The second line contains n integers representing $a_1, a_2, \ldots, a_n$.

Constraints: $1 \leq n \leq 10^5, 0 \leq a_i \leq 10^5$.

Output Format

Output a single number, representing the value of the answer modulo 998244353.

Examples

Input	Output
2 1 2	26

C: Hospital Blood Bank

Time limit: 2.0 sec

Memory limit: 512 MB

There are specific antigens in human blood, and if a patient's blood does not contain the corresponding antigen, they cannot receive blood that contains it.

For example, when blood is divided into four types: `A`, `B`, `AB`, `O`. `A` blood has antigen `A`, `AB` blood has both `A` and `B` antigens. Specifically, `O` blood has no antigens. Therefore, an `A` blood type patient can only receive blood type `A` or `O`, a `B` blood type patient can only receive blood type `B` or `O`, an `O` blood type patient can only receive `O` blood, and an `AB` blood type patient can receive any of the four types mentioned above.

You unexpectedly landed on a planet where there may be 25 different antigens in the blood (represented by uppercase letters other than `O`, each uppercase letter representing an antigen, if a person's blood does not contain any antigens, it is represented by `O`), and the complex blood mechanism has led to a very tight blood supply on this planet. Due to recent disasters, they have a large number of patients in need of blood transfusions and hope to use as much blood as possible from the current blood bank to assist suitable patients. Unable to

find a solution, they found you and hoped that you, who came from afar, could help them solve this problem.

Input Format

The first line of input contains two positive integers separated by a space, n and m ($1 \leq n \leq 1000, 1 \leq m \leq 1000$), indicating the number of blood types in the blood bank and the number of blood types required.

The next n lines each contain a string and a positive integer ($\leq 10^6$) separated by a space, representing the blood type information and the remaining quantity of one blood type in the blood bank.

The next m lines each contain a string and a positive integer ($\leq 10^6$) separated by a space, representing the blood type information and the quantity required for one blood type.

Constraints:

Each string in the input contains only uppercase letters, and each string does not contain duplicate letters. Only strings of length 1 may contain `O`.

There may be duplicate stock and demand blood types in the input.

Output Format

Output a single line containing a non-negative integer, indicating the maximum number of units of blood in the blood bank that can satisfy patient demand.

Example

Input	Output
4 4 O 50 A 36 B 11 AB 8 O 45 A 42 B 10 AB 3	99
3 4 O 2 EJFIDKM 7 O 3 O 5 MFJKIED 1 ABCDEFGHIJKLMNPQRSTUVWXYZ 5 ZYXWVUTSRQPNMLKJIHGFEDCBA 1	12

Hint

In the first example, the extra 5 units of `O`-type blood can be used for `A`-type blood patients, totaling $45 + 41 + 10 + 3 = 99$ units of demand that can be met.

D: DDL Warriors

Time limit: 1 second,
Memory limits: 512 MB

Frank is a competitive programming enthusiast who has spent countless late nights secretly training, and eventually achieved good results in regional contests.

However, after Frank finished today's training, he found that some tasks assigned by his professor were nearing the deadline! Despite being physically and mentally exhausted, Frank thought of the three DDL warriors of the Binary Indexed Tree (BIT) who, through their unremitting efforts, achieved a good ranking of 13th place in the world finals. Suddenly, he felt full of strength!

Frank has n tasks, and he has a total of m minutes to complete these tasks. Specifically, completing the ith task requires t_i minutes, and completing this task completely will increase the professor's satisfaction by w_i. If the task is completed in less than t_i minutes, the professor's satisfaction will decrease by c_i for each minute less than t_i, until it reaches 0. In other words, if x minutes $(0 \leq x \leq t_i)$ are spent completing the ith task, the professor's satisfaction will increase by $max(0, w_i - c_i * (t_i - x))$.

Frank wants to maximize the professor's satisfaction, so he requests the maximum value of this satisfaction.

Input

The first line contains a positive integer $T(1 \leq T \leq 1000)$, indicating the number of test cases.

For each test case, the first line contains two positive integers $n, m(1 \leq n, m \leq 5000)$, representing the number of tasks and the time Frank has to complete them.

The next n lines each contain three positive integers $t_i, w_i, c_i(1 \leq t_i \leq m, 1 \leq c_i \leq w_i \leq 10^9)$, as described above.

It is guaranteed that the sum of n and m for all test cases will not exceed 10^4.

Output

For each test case, output a positive integer representing the maximum satisfaction.

Example

Input	Ouput
3 3 7 3 9 3 2 10 5 6 6 1 2 3 3 13 6 3 14 5 2 1 1 13 1 1 15 1	 21 14 27

Note

For the first test case, Frank can spend 3 minutes completing the first task, earning 9 satisfaction points; then spend 2 minutes completing the second task, earning 10 satisfaction points; and finally spend 2 minutes completing the third task, earning 2 satisfaction points, for a total of $9 + 10 + 2 = 21$ satisfaction points.

For the third test case, Frank can spend 1 minute completing the first task, earning 13 satisfaction points; although Frank did not spend time on the first task, he can still earn 14 satisfaction points, for a total of $13 + 14 = 27$ satisfaction points.

E: Misumi Aoi's Afternoon Tea Time

Time limit: 1 second, Memory limit: 512 MB

Misumi Aoi often enjoys wonderful afternoon tea time at the 1906 Cafe at USST, and she is very knowledgeable about coffee-related knowledge.

Due to the upcoming "Lenovo Cup" at USST in the second semester of the year, Misumi plans to develop a new coffee with the help of Mafuyu, a staff member at 1906, to cheer on the participants.

Misumi and Mafuyu now have n types of coffee beans, where the ith type of coffee bean has a magical power value of a_i. Initially, each type of coffee bean forms its own pile.

We define the "Grace" value of a pile of coffee beans as the Minimum Excluded Number (MEX) of the magical power values of the coffee beans in that pile. Here, MEX represents the smallest natural number that does not appear in the set. For example, if a pile of coffee beans contains three types of coffee beans with magical power values of $0, 1, 3$, then the "Grace" value of this pile of coffee beans is $MEX(0, 1, 3) = 2$.

They can perform the following operations several times:

- Mix any two or more piles of coffee beans, which will increase the deliciousness of the coffee by the sum of the "Grace" values of each pile of coffee beans. For example, if there are three piles of coffee beans with deliciousness values of $[0, 1, 2]$, $[0, 1, 4]$, $[1, 2]$, then merging these three piles of coffee beans can obtain a deliciousness value of $Grace(0, 1, 2) + Grace(0, 1, 4) + Grace(1, 2) = 3 + 2 + 0 = 5$.

Their goal is to merge all coffee beans into one pile and maximize the sum of the deliciousness values generated during the merging process. Please help them find the maximum sum of the deliciousness values generated during the merging process.

Input

The first line contains a positive integer $T(1 \leq T \leq 10^5)$, indicating the number of test cases.

For each test case, the first line contains a positive integer $n(1 \leq n \leq 10^6)$, representing the number of types of coffee beans.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n(0 \leq a_i \leq n - 1)$, representing the magical power values of each type of coffee bean.

It is guaranteed that the sum of n for all test cases will not exceed 10^6.

Output

For each test case, output a positive integer representing the maximum sum of the deliciousness values generated during the merging process.

Example

Input	Output
3 3 0 0 0 4 2 1 2 0 4 3 2 2 1	4 6 0

Note

For the first test case in the example, one feasible mixing method is as follows:

First, mix the first pile of coffee beans with the second pile of coffee beans, which will gain a deliciousness value of $MEX(0) + MEX(0) = 1 + 1 = 2$.

Then, mix the resulting pile of coffee beans with the third pile of coffee beans, which will gain a deliciousness value of $MEX(0, 0) + MEX(0) = 1 + 1 = 2$.

The total deliciousness value obtained will be 4. It can be proven that there is no other mixing method that can obtain a larger deliciousness value.

For the third test case in the example, since no matter how the coffee beans are merged, the deliciousness value obtained will be 0, simply mix all the coffee directly into one pile.

F: The Reign of the Flame Dragon over the Continent of Teyvat

Time limit: 2.5 seconds
Memory limit: 2048 MB

The continent of Teyvat is governed by the gods. It has n cities connected by $n-1$ bidirectional roads. Each city belongs to a camp of gods, and there may be multiple cities belonging to the same camp.

However, the Flame Dragon intends to overthrow the rule of the gods over the continent of Teyvat and establish a kingdom. Specifically, the Flame Dragon needs to choose a connected block of cities, and all the cities in the connected block will become part of the kingdom and withdraw from their camps.

The Flame Dragon wants to stabilize the continent of Teyvat as much as possible. The instability value of the kingdom is defined as the distance between the two farthest cities among the cities belonging to the kingdom. The instability value of the remaining cities is defined as the number of camps of the remaining cities. The total instability value of the continent of Teyvat is the maximum of these two instability values.

The Flame Dragon wants to minimize the instability value of the continent of Teyvat. Please help the Flame Dragon find this minimum value.

Input

The first line contains two integers $n(1 \leq n \leq 100000), m(1 \leq m \leq 2000)$, indicating the number of cities and the number of camps in the continent of Teyvat.

The next line contains n integers. The ith integer $a_i(1 \leq a_i \leq m)$ represents the camp to which the ith city belongs. It is guaranteed that each camp has at least one city, and each integer in the range $[1, m]$ appears at least once.

The next $n-1$ lines each contain two integers $u, v(1 \leq u, v \leq n, u \neq v)$, indicating that there is a bidirectional edge between city u and city v.

Output

Output an integer, representing the minimum instability value.

Example

Input	Output
5 3 1 2 3 2 1 1 2 1 3 3 4 2 5	2

Note

The Flame Dragon can choose to establish his kingdom in city 3. In this case, the distance between the farthest cities in the kingdom is 0. Among the remaining cities, there are cities belonging to camps 1 and 2, and the instability value of the remaining cities is 2. The total instability value of the continent of Teyvat is $max(0, 2) = 2$.

G: Basic Substring Structure

Time limit:	1 second
Memory limit:	1024 megabytes

After writing the paper *Faster Algorithms for Internal Dictionary Queries*, Little Cyan Fish and Kiwihadron decided to write this problem.

Let $\mathrm{lcp}(s,t)$ be the length of the longest common prefix of two strings $s = s_1 s_2 \dots s_n$ and $t = t_1 t_2 \dots t_m$, which is defined as the maximum integer k such that $0 \le k \le \min(n,m)$ and $s_1 s_2 \dots s_k$ equals $t_1 t_2 \dots t_k$.

Little Cyan Fish gives you a non-empty string $s = s_1 s_2 \dots s_n$. Let $f(s) = \sum_{i=1}^{n} \mathrm{lcp}(s, \mathrm{suf}(s,i))$, where $\mathrm{suf}(s,i)$ is the suffix of s starting from s_i (i.e. $\mathrm{suf}(s,i) = s_i s_{i+1} \dots s_n$). Note that in this problem, the alphabet contains n letters, not just 26.

For each $i = 1, 2, \cdots, n$, you are asked to answer the following query: if you MUST change s_i to another different character c ($c \neq s_i$), choose the best character c and calculate the maximum value of $f(s^{(i)})$, where $s^{(i)} = s_1 \dots s_{i-1} c s_{i+1} \dots s_n$.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \le n \le 2 \times 10^5$) indicating the length of the string.

The second line contains n integers $s_1, s_2, \dots, s_n$ ($1 \le s_i \le n$) where s_i indicates that the i-th character of the string is the s_i-th letter in the alphabet.

It's guaranteed that the sum of n over all test cases doesn't exceed 2×10^5.

Output

Let $m(i)$ be the maximum value of $f(s^{(i)})$. To decrease the size of output, for each test case output one line containing one integer which is $\sum_{i=1}^{n} (m(i) \oplus i)$, where $\oplus$ is the bitwise exclusive or operator.

Example

standard input	standard output
2 4 2 1 1 2 12 1 1 4 5 1 4 1 9 1 9 8 10	15 217

Note

For the first sample test case, let's first calculate the value of $m(1)$.

- If you change s_1 to 1, then $f(s^{(1)}) = 4 + 2 + 1 + 0 = 7$.
- If you change s_1 to 3 or 4, then $f(s^{(1)}) = 4 + 0 + 0 + 0 = 4$.

So $m(1) = 7$.

Similarly, $m(2) = 6$, $m(3) = 6$ and $m(4) = 4$. So the answer is $(7 \oplus 1) + (6 \oplus 2) + (6 \oplus 3) + (4 \oplus 4) = 15$.

H: Turn on the Light 2

Time limit:	1 second
Memory limit:	1024 megabytes

The much-anticipated Universal Cup Final is around the corner! Little Cyan Fish is busy preparing the venue for the competition. To make the venue gorgeous, Little Cyan Fish is planning to hang some light bulbs.

Little Cyan Fish has m wires and plans to connect n light bulbs using these wires. Each wire needs to connect two distinct bulbs and make all the bulbs form a single connected component. To maintain safety, for any two bulbs there can be at most one wire connecting them directly, and no more than d wires can be attached to a single bulb.

Once the bulbs are connected, Little Cyan Fish wants to illuminate some of them. Given that active bulbs generate heat, positioning two lit bulbs adjacent to each other could be dangerous. Therefore, if two bulbs are directly connected by a wire, they must not be illuminated at the same time. On the other hand, he doesn't want too few lights on, so he does not want to see an unlit bulb, such that all bulbs directly connected to it are also unlit.

This leads Little Cyan Fish to wonder about the number of different plans to turn on those bulbs to fulfill the requirements. More so, he is keen on finding the optimal way to connect all the light bulbs to maximize the number of the feasible lighting plans.

Given the integers m and d, your goal is to assist Little Cyan Fish in determining the optimal way to connect n bulbs using all m wires, with the intent of maximizing the number of the ways to illuminate the bulbs. Note that you have to determine the value of n by yourself.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 200$), indicating the number of test cases. For each test case:

The first and only line contains two integers m and d ($2 \le m \le 20$, $2 \le d \le m$).

Output

For each test case:

In the first line, output an integer w ($1 \le w \le 2^{m+1}$), indicating the maximum number of ways to illuminate the bulbs.

In the second line, output an integer n ($1 \le n \le m+1$), indicating the number of bulbs Little Cyan Fish should use.

Then output m lines, where the i-th line contains two integers u_i and v_i ($1 \le u_i, v_i \le n, u_i \ne v_i$) separated by a space, indicating a wire connecting the u_i-th and the v_i-th bulb.

Example

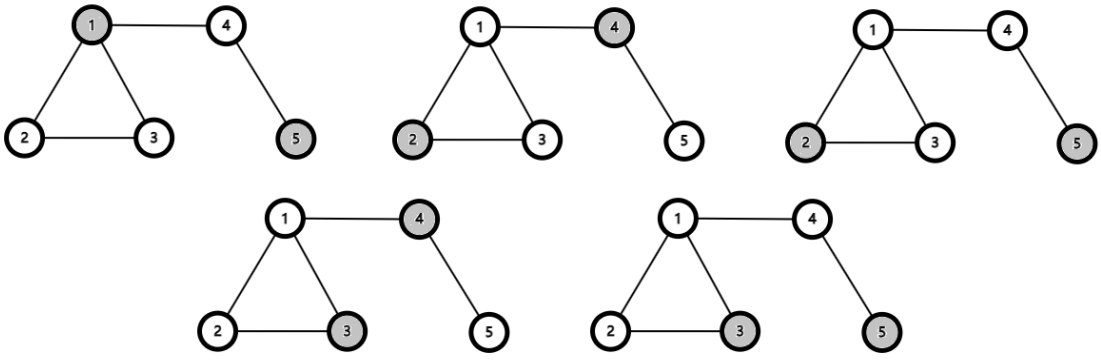
standard input	standard output
3 2 2 5 4 6 2	2 3 1 2 2 3 5 5 1 2 1 3 2 3 1 4 4 5 7 7 1 2 2 3 3 4 4 5 5 6 6 7

Note

We use colored circles to represent lit bulbs.

For the first sample test case, the 2 ways to illuminate the bulbs are illustrated below.

For the second sample test case, the 5 ways to illuminate the bulbs are illustrated below.



I: New Level

Time limit: 2 seconds
Memory limit: 512 megabytes

Robocity has n crossroads connected by bidirectional roads. There are m roads in total, and all crossroads are reachable from each other. There is a level assigned to each crossroad specified by a number from 1 to k, inclusive. Any pair of crossroads directly connected by a road has distinct levels.

The city leaders are planning a reform. Namely, they want to assign new levels to crossroads, so that each level still has a value from 1 to k, connected crossroads would have different levels, and an additional condition has to be met: for each pair of crossroads u and v there must exist a path between them, such that any two adjacent crossroads along it have levels that differ by 1 modulo k.

Formally, for each pair of crossroads (u, v) there should exist a sequence of crossroads $p_1, \ldots, p_l$, such that:

- $p_1 = u$;
- $p_l = v$;
- for each i from 1 to $l - 1$, crossroads p_i and p_{i+1} are connected, and either their levels differ by one, or one of them has level of 1 and another has level of k.

Robocity government is convinced that such level assignment exists and asks you to find it.

Input

The first line contains three integers n, m, k ($1 \le n, m, k \le 500\,000$), number of crossroads, roads, and levels.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le k$), c_i is the level of the crossroad i.

Then m lines follow, each of them contains two integers u, v ($1 \le u, v \le n$; $u \ne v$), a pairs of crossroads connected by a road.

It is guaranteed that there are no two roads connecting the same pair of crossroads, and that there exists a path between each pair of crossroads.

Output

Output n integers $d_1, d_2, \ldots, d_n$ ($1 \le d_i \le k$), the levels of the crossroads in the new assignment.

Example

standard input	standard output
4 4 4 1 2 3 1 1 2 1 3 2 3 3 4	4 3 2 1