Problem A. AC

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

Crystal's fortune predict system is successfully developed!

The predict system is a distributed system consists of N computers. When it receives a predict request, each computer will generate one lowercase letter as its output. The final fortune predict result is determined by the concentration of all N outputs.

Tired of getting bad predictions like `awful: millions of bugs`, Ben decides to hack into the predict system and modify the predict result. He has already got the access permission of every computer in the predict system, so he can modify their output to any letter arbitrarily.

As Ben is going to take part in ICPC Asia Regional Kunming Site 2077, he wants predictions like `suitable for writing codes` or `will get accepted for every problem`. He has found that the more times the substring `ac` occurs in the concentration of all N outputs, the luckier he will get in the contest. But as the contest is coming soon, he only has time to modify at most K outputs of the computers in the predict system.

As Ben is busy hacking into the system, could you tell him how to get the most `ac` substrings after his modification?

Input

The first line contains two integers N and K ($1 \le N \le 5 \times 10^5, 0 \le K \le N$).

The second line contains a string of length N, denoting the origin prediction. It is guaranteed that the string consists of lowercase English letters.

Output

Output two lines. The first line contains a single integer, denotes the maximum number of `ac` substring Bob can get, after his modification. The second line contains the final modified predict string. If there are multiple ways that results in the maximum number of `ac` substring, print any.

Example

standard input	standard output
9 2 arakbacca	3 acacbacca

Problem B. Chessboard

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Your are playing a computer game. The game is played on a chessboard of n rows and m columns.

For each grid on the board, you can choose to put a black piece or a white piece on it, or leave it blank. Denote (i, j) as the grid in the i-th row and the j-column. For each $i \in [1, n], j \in [1, m]$, putting a black piece on (i, j) earns you $sb_{i,j}$ points, while a white piece earns you $sw_{i,j}$ points, and leaving blank do not affect your score. The overall score will be the sum of the points earned by each piece. Note that a grid can contain at most **one** piece at the same time, that is, you cannot put a white piece and a black one simultaneously. It is guaranteed that $sb_{i,j}$, $sw_{i,j}$ are all non-negative integers.

After you finish placing the pieces, the computer program checks whether the pieces are put in a **beautiful** way or not. Let's define b_i as the number of black pieces in the i-th row, B_i as the number of black pieces in the i-th column, w_i as the number of white pieces in the i-th row, and W_i as the number of white pieces in the i-th column. The pieces are considered **beautiful** if

- For any $i \in [1, n]$, $b_i - w_i \in [l_i, r_i]$ holds.
- For any $i \in [1, m]$, $B_i - W_i \in [L_i, R_i]$ holds.

Tired of high scores, you decide to **minimize** your score. To simplify the problem, you only need to output the minimum possible score among all beautiful placements of pieces.

Input

The first line of input contains two integers $n, m (2 \le n, m \le 50)$, denoting the number of rows and columns of the board.

The next n lines describe the points earned by putting black pieces. The i-th of them contains m integers, the j-th of which denotes $sb_{i,j} (0 \le sb_{i,j} \le 10^3)$.

The next n lines describe the points earned by putting white pieces. The i-th of them contains m integers, the j-th of which denotes $sw_{i,j} (0 \le sw_{i,j} \le 10^3)$.

The next n lines describe the constraints on each row, the i-th of which contains two integers $l_i, r_i (-m \le l_i \le r_i \le m)$, whose meaning can be found in the statement above.

The next m lines describe the constraints on each column, the i-th of which contains two integers $L_i, R_i (-n \le L_i \le R_i \le n)$, whose meaning can be found in the statement above.

It is guaranteed that there exists at least one strategy that satisfies all the constraints.

Output

Output a single integer, denoting the **minimum** score you can get.

Example

standard input	standard output
3 3 6 9 0 3 2 7 6 4 6 0 6 5 1 6 9 8 5 7 -3 -1 -3 0 1 3 0 0 1 1 -2 0	9

Problem C. Cities

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

Bob lives in a chaotic country with n cities in a row, numbered from 1 to n. These cities are owned by different lords, and the i-th cities currently belongs to the a_i-th lord. To simply problems, we assume there are n lords in the country, and they are also numbered from 1 to n. Some lords may take control of multiple cities, while some new-born lords have not got any cities yet.

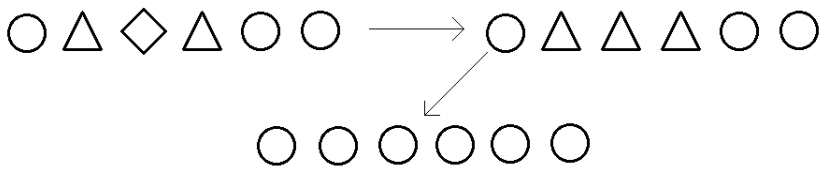
Obviously, the greedy lords are not satisfied with the number of territories they have, so the country is constantly at war. Bob wants to change that, by making all the cities belong to the same lord!

Bob can perform some magical operations to support his grand plan. With the help of each magic, Bob can do the following:

- Choose some cities with consecutive indices such that they belong to the same lord, and assign them to any other lord.

As magics are really tiring, Bob wants to know the minimum number of such operations he needs to use to make all the cities belong to one lord.

The following picture shows an example where $n = 6$. Different shapes are used to represent cities belonging to different lords. As shown in the picture, the minimum number of magic operations used is 2.



Input

The first line contains a single integer t ($1 \le t \le 160$) — the number of test cases.

The first line of each test case contains an integers n ($1 \le n \le 5000$) — the number of cities in the country.

The second line of each test case contains n integers a_i ($1 \le a_i \le n$)— the i-th city was originally owned by the a_i-th lord. It is guaranteed that **currently no Lord will have more than 15 cities, which means no one a_i will appear more than 15 times in this line.**

It is guaranteed that the sum of n over all test cases doesn't exceed 6000.

Output

For each test case, print a single integer indicating the answer.

Example

standard input	standard output
2 8 4 3 1 2 1 1 3 3 5 1 2 3 2 1	3 2

Problem D. Competition Against a Robot

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

As a tradition, the Code Village holds an annual championship, named ICPC (Intelligence Championship for Platinum Coders) to search for its most outstanding genius. The participants take part in various contests to find out who is the potential genius. The competitions include Go, Reversi, Sudoku and even Paper-Scissor-Rock, and it is fascinating to see the unpredictable result. Well... At least a few years ago.

Everything changed when an AI robot came to the village - it seems weird, but as the name of the contest shows, anything related to 'intelligence' can take part freely. The robot soon won all the competitions without difficulty, and remained the defending champion for years and years. Everyone considered it to be undefeatable. And this time, your task will be - as you have figured out - beat the robot!

You are not alone - you will have a teammate. To make the game more interesting, the organizing committee has slightly modified the rule and it will be a guessing game. The game process is as follows:

- The judges announce two positive integers n, k to the three players - you, your teammate, and the robot.
- The robot generates an integer sequence of length n - let's call it T - and selects a secret integer $p \in [0, n)$. Each element of T should fall in the range $[0, k)$, that is, each $T_i (0 \leq i < n)$ should be a non-negative integer and strictly less than k. Then it submits T and p to the judges. (Note that in this problem all the sequences are 0-indexed.)
- You receive the sequence T and the number p from the judges, and your task is to tell the exact value of p to your teammate. To achieve that, you **must** select exactly **one** index $j \in [0, n)$, and set T_j to $(T_j + 1) \bmod k$.
- Your teammate then receives the sequence T, and has to figure out the hidden integer p. Note that he doesn't have access to the original string generated by the robot; the only information he gets is the string that you write. He has only one chance to submit his answer, and if he succeeds, both of you win the game.

Some typical settings may apply here. For example, the three of you all have unlimited and accurate memory, your team can discuss strategies before the game starts (of course NOT during the game), while the robot has the access to your strategy. You can also assume that the three of you will play optimally to achieve their goals: for you two the goal is definitely to win the game, while for the robot it will do its best to prevent your victory.

Let's take some examples. For $n = 1$, your team will definitely win by always answering $p = 0$. For $n = 2, k = 2$, a solution that ensures your victory works like this: your friend always answer the value of T_0, and when you enter the room you just check if T_0 differs from p: if so then choose $j = 0$, otherwise choose $j = 1$. It is clear to see that such strategy works perfectly. However, it can be proved that for $n = 3, k = 2$ your team has no chance to win at all.

There are five hours left before the competition, and you decide to do some practice by writing a program to determine that for certain pair of (n, k), which side will win the game. You will have to answer Q independent queries.

Input

The first line of input contains an integer $Q (1 \leq Q \leq 10^5)$, the number of queries you have to answer.

Each of the next Q lines indicates a query. The i-th among them contains two integers $n_i, k_i (1 \leq n_i, k_i \leq 10^{18})$, denoting the i-th situation with parameters $n = n_i, k = k_i$. Recall that each query is independent.

Output

Output Q lines, the i-th of which shows the winning side of the i-th situation. If your team is going to win, print a line of `HUMAN`; otherwise display `ROBOT`.

Example

standard input	standard output
3 1 10 2 2 3 2	HUMAN HUMAN ROBOT

Problem E. Counting Binary Trees

Input file:	standard input
Output file:	standard output
Time limit:	3 seconds
Memory limit:	256 megabytes

You are given an array of positive integers $k_1, k_2, \ldots k_m$.

A binary tree is **special** if all of these conditions are satisfied :

- 1. A positive integer is written on every node of the tree.
- 2. Every non-leaf node has both left child and right child, and the number written on it is equal to the product of numbers written on left child and right child. Here, we define a leaf as a node with no children, and a non-leaf node is a node that is not a leaf.
- 3. The number written on any leaf is a multiple of at least one k_i. For two integers x, y, we consider x to be a multiple of y if there exists some integer z such that $x = yz$.

Two binary trees are different if one of these conditions is satisfied :

- 1. The numbers written on their root are different.
- 2. Their left subtrees are different or their right subtrees are different.

Note that the definition above is recursive.

For a given n, you need to find the number of special binary trees whose number written on the root is not greater than n. Since the answer can be quite large, output it modulo 998244353.

Input

The first line contains a single integer T ($1 \le T \le 10$) — the number of test cases. Then T test cases follow.

The first line of each test case contains two integers n, m ($2 \le n \le 10^9, 1 \le m \le 4$).

The second line of each test case contains m integers $k_1, k_2, \ldots, k_m$ ($2 \le k_i \le 100$).

Output

For each test case, print a single integer: the number of special binary trees whose number written on the root is not greater than n. Remember that you only need to print it modulo 998244353.

Example

standard input	standard output
2 6 2 2 3 100 2 6 9	7 28

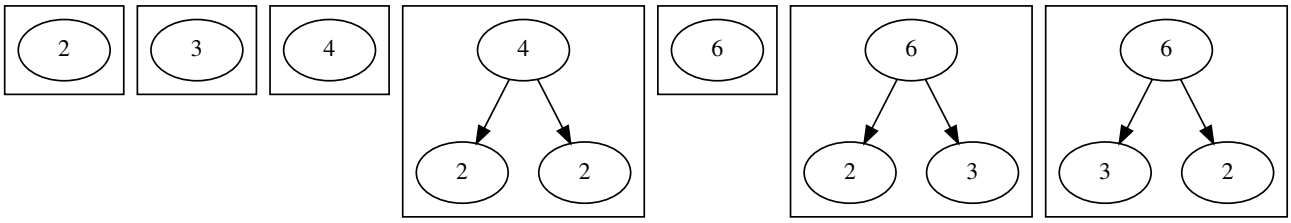
Note

In the first test case, you can find:

- 1 special binary tree whose root is 2;

- 1 special binary tree whose root is 3;
- 2 special binary trees whose root is 4;
- 3 special binary trees whose root is 6.

So the answer is $1 + 1 + 2 + 3 = 7$. Here is an illustration for the 7 trees.



Problem F. Generating Strings

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

Mr. Chaos is a high school student. Teachers always ask him to write a lot of compositions, but he isn't interested in it at all. Tired of generating meaningless words by himself, he invented a machine to help him.

His teacher has posted the model article – string S , on the blackboard. His machine can generate any lowercase string T of length n at a time. For a fixed string T, the scoring method is simple: for any pair of integers $[l, r]$ such that the substring $T[l, r]$ is a palindrome, the final score will be added by the number that $T[l, r]$ occurs in S. Formally, let's define $T[l, r](1 \le l \le r \le n)$ as the string $T_l T_{l+1} ... T_r$, a string is **palindromic** if it remains the same string when reversing, and the **occurrence time** of P in S, $\mathrm{OCC}_S(P)$, is the number of pairs of $[l, r]$ such that $S[l, r] = P$. Then, the value of T, namely $\mathrm{V}(T)$, is defined as

$$\mathrm{V}(T) = \sum_{1 \le l \le r \le n, T[l,r] \text{ is palindromic}} \mathrm{OCC}_S(T[l, r])$$

For example, given $S =$ `bbbaa`, when the machine generated a string $T =$ `abbaabbaa`, one of its palindromic substrings P will be `bb`. $\mathrm{OCC}_S(P)$ is 2 because `bb` occurs in S twice. Note that there are two `bb` in this case, and they are considered as different substrings and calculated twice.

As Mr. Chao's best friend, you are asked to tell him the sum of value of all the T that his machine can generate.

You accept this mission without thinking twice. However, it seems that you are getting into trouble, because the teacher makes m revisions on string S. Each time the last character will be deleted or a new character c will be added to the end. So now you have to answer the question for $m + 1$ times.

Formally, let $\mathcal{T}$ be the set of all strings of length n that only contain lowercase English letters. You are going to calculate this formula

$$\sum_{T \in \mathcal{T}} \mathrm{V}(T)$$

at the beginning and after each revision of S, that is, in total $m + 1$ times. As the answer may be too large, you only need to output its remainder modulo $10^9 + 7$

Input

The first line contains one integer $t(1 \le t \le 25)$ — the number of test cases. Then t test cases follow.

The first line of each test case contains two integers and one string : n, m, S $(1 \le n, m, |S| \le 5 \times 10^5)$ — the length of the string that machine can generate, the number of revision and the original article S. It is guaranteed that S only contain lowercase English letters.

Each of the next m lines indicates a revision, in the order that they are made. The format will either be `1` c, which means that the character c is added to the end of S, or `2`, which means that the last character of S is deleted. It is guaranteed that c is a lowercase English letter.

It is guaranteed that in a single test file, the sum of n, the sum of m, and the sum of the length of S among all test cases do not exceed 2×10^6.

Output

For each test case print $m+1$ lines, each containing a single integer. The first line should be the answer of

the original S, while the next m lines should be the answer after each revision. Note that you only need to print them modulo $10^9 + 7$.

Example

standard input	standard output
3 6 3 aaa 2 1 b 1 a 5 3 aabaaa 2 2 2 6 2 aababa 1 a 1 b	218504832 144861392 216149648 287508208 13924249 11567037 9211852 6924944 430225380 503798516 575088800

Problem G. Gift

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

EQWE is a pastry chef who has N friends. This year (obviously, the year 2021), he wants to give each friend a birthday cake made by himself.

If EQWE wants to give the i-th friend a birthday cake made by himself, he need c_i days (do NOT need to be contiguous) before his friend's birthday to make it. After that he will get v_i favorable impression. Note that it will be OK for him to finish the cake exactly on the birthday, as birthday parties are always held at night.

But EQWE doesn't have much time and he has another plan. There are M special gifts in the shop. EQWE can pay a_j yuan to get the j-th special gift and send it to any friend. Note that each gift is unique, which means that he can buy each gift at most once. After that he will get b_j favorable impression.

EQWE's friends are very polite, so they do NOT want to receive more than one gift (including birthday cakes and special gifts). Note that to gain some favorable impression from a friend, the selected gift must be sent on the exact date of the friend's birthday.

Suppose that it is the first day of the year 2021 now (and he can start making birthday cakes immediately), and EQWE has W yuan. What's the maximum number of favorable impression that EQWE can get in 2021?

Input

The input file starts with an integer $T(1 \le T \le 100)$, denoting the number of test cases. Then T test cases follow.

For each test case, the first line contains three integers $N(1 \le N \le 500), M(1 \le M \le 15), W(1 \le W \le 10^4)$, denoting the number of friends, the number of special gifts, and the amount of money that EQWE has.

Each of the following N lines describes a friend, in the format of $year - month - day\ c_i\ v_i$, where $year - month - day$ is the date of the friend's birthday, $c_i(1 \le c_i \le 30)$ is the day needed to make the birthday cake for the friend, and $v_i(1 \le v_i \le 10^6)$ is the favorable impression that EQWE can get. It is guaranteed that c_i, v_i are integers. It is also guaranteed that the date given are all valid dates between the year 1990 and 2010, that is, $year$ is an integer between 1990 and 2010, $month$ is an integer between 1 and 12, and day is a positive integer which do not exceed the number of days in the given month.

The next M lines describe the special gifts, the i-th of which contain two inte The i^{th} line is $a_i\ b_i(1 \le a_i \le W, 1 \le b_i \le 10^6)$

Output

For each test case, output a line containing a single integer, denoting the maximum number of favorable impression that EQWE can get.

Example

standard input	standard output
1 2 2 100 2000-01-01 10 13 2000-12-31 30 92 99 46 2 2	138

Note

It is commonly known that a year is divide into 12 months, and for the most of the time the numbers of days in each month are 31,28,31,30,31,30,31,31,30,31,30 and 31. The only exception is that for leap years the second month contains 29 days. Of the time range mentioned in the problem (that is, from 1990 to 2021), the leap years are 1992, 1996, 2000, 2004, 2008, 2012, 2016 and 2020.

Problem H. Hard Calculation

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

Hooray! It is the first time that Kunming holds an ICPC regional contest. Suppose that everything goes on well and the Kunming Regional Contest is held each year. In which year will the x-th Kunming Regional Contest be held?

Input

The first and only line of input contains a single integer $x(1 \leq x \leq 100)$.

Output

Output a single integer, denoting the year when the x-th Kunming Regional Contest will be held.

Examples

standard input	standard output
1	2021
100	2120

Note

Note that it is the year 2021 right now.

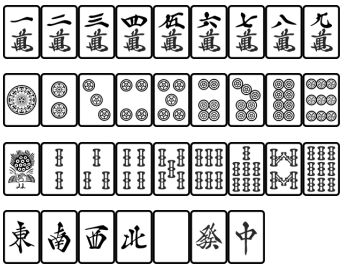
Problem I. Riichi!!

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

Yui is a cute girl expert in a kind of `Mahjong` game. In case you are not familiar with the game, here we briefly introduce its rules.

`Mahjong` is played with **tiles**, divided into 34 **kinds**. To simplify the problem, we assume that **there is an infinite number of every kind** (although in real-world game one kind usually contains up to 4 tiles). The 34 kinds of tiles can be further divided into 4 **suites**, named as `bing`, `suo`, `wan`, and `zi`. The `bing`, `suo`, `wan` have 9 kinds for each suite and `zi` tiles has only 7 kinds.

Here are the all 34 kinds of tiles used in `Mahjong` game: Each row refers to a suite of tiles — `suo`, `bing`, `wan`, `zi` in order.

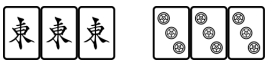


Minor differences exist in various versions of `Mahjong` game, and here we only consider some basic rules.

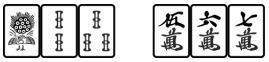
During the game, each player holds 13 tiles in hand. For each round, one player would draw one tile from the Mahjong table, then discard one tile from one of the 14 tiles owned at the time. A player wins the game if in a round he can create a **special combination** (defined below) with the 13 tiles in hand and 1 tiles drawn from table or discarded from other player.

A **special combination** consists of 14 tiles, which can be divided into four `kezi` or `shunzi`, and an additional `quetou`.

Here, `kezi` is a set of 3 identical tiles:

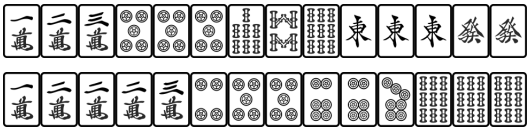


And `shunzi` is a set of 3 continuous tiles (please aware that suite `zi` cannot form `shunzi`) :

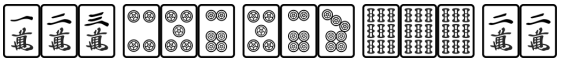


Finally, `quetou` is a pair of identical tiles:

Here are some samples of special combinations:

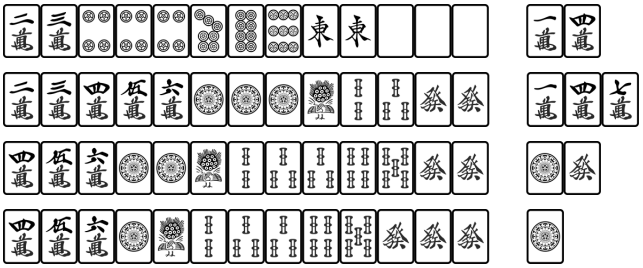


The second example might be confusing. Since you can combine your tiles arbitrary, the actual combination is the follows:



If the player can obtain a special combination by getting some tile, then we call the player is in the `Riichi` status. In this status one can achieve victory once he/she gets the desired kind of tile!

For example, the following sets of 13 tiles are in the `Riichi` status, the tiles after the space indicates the desired tiles to achieve victory.



Now Yui is playing the `Mahjong` game. Sometimes she is very close to victory that if she tosses some tile she would reach the `riichi` status, while sometimes such tiles does not yet exist. And sometimes she has already got a winning hand of 14 tiles!

To help you improve your `Mahjong` skills, Yui decides to give you a test. Now it is Yui's turn and she has 14 tiles in her hand. Please tell her which tile should be discarded in order to reach `riichi` status, or just tell her that she has already won in this round!

Input

The input contains multiple tests cases. The first line includes a single integer T — the number of test cases. It is guaranteed that $T \leq 10,000$.

Each of the next T lines indicates a test case. It contains a string s of 28 characters, describing the 14 tiles that Yui currently has. For every $1 \leq i \leq 14$, the i-th tile obtained by Yui is described by the $(2i-1)$-th and $2i$-th characters in the string: the former is a digit denoting the rank of the tile in its suite and the latter is one of `w, b, s, z`, which means the suite `wan, bing, suo` and `zi` respectively. It is guaranteed that all the s in the input are valid and legal.

Output

Output the answer for each test case separately. For each test case, if Yui has already reached the winning status in this round, output `Tsumo!` in a single line.

Otherwise, output a single integer *ans* in a single line, the number of choices for discarding tiles to reach `riichi` status.

Each of the next *ans* lines should indicate one way to reach the `riichi` status. It should start with two characters indicating the tile to be discarded. To prove that such way leads to a `riichi` status, you should print all the tiles that can lead to victory for the status. For clarity, print a space between the first two characters and the rest of the line.

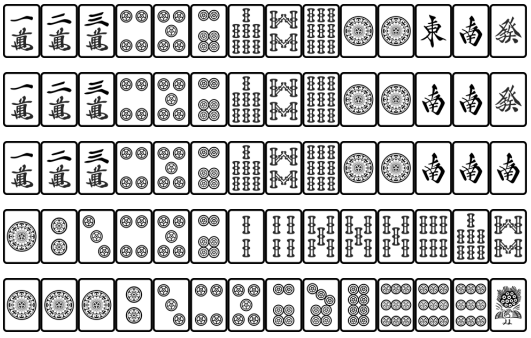
Important: Pay attention to the order when printing the desired tiles, as there is NO special judge. For tiles in different suites, print them in the order `wan`, `bing`, `suo`, `zi` (that is, always print `wan` tiles at first, and so on). For tiles in the same suite, print the cards in the ascending order of their digits (that is, the tile with smaller number goes first). Additionally, if there are several tiles available for a `riichi` status to achieve victory, you should also sort them in the same way. See the Sample Output for details.

Example

standard input	standard output
5 1w2w3w4b5b6b7s8s9s1b1b1z2z6z 1w2w3w4b5b6b7s8s9s1b1b2z2z6z 1w2w3w4b5b6b7s8s9s1b1b2z2z2z 1b2b3b4b5b6b2s4s5s5s5s6s7s8s 1b1b1b2b3b4b5b6b7b8b9b9b9b1s	0 1 6z 1b2z Tsumo! 4 2s 3s4s6s9s 4s 2s 5s 3s 8s 3s 4 2b 1s 5b 1s 8b 1s 1s 1b2b3b4b5b6b7b8b9b

Note

The samples are the tiles below:



Problem J. Mr. Main and Windmills

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

Mr. Main took a train from city s to city t and passed a plain full of windmills. The train ran in a straight line. A windmill is a machine used for wind power generation. Its fan blades rotate when the wind blows. From his perspective, colorful windmills lined up on the horizon from left to right.

As the train was running, the order of windmills from his perspective was constantly changing: a windmill was originally on the left/right of another, and then changed to its right/left;

Given the coordinates of the windmills, please find the coordinate of him when he just observed the h-th windmill exchanged order with other windmills for the k-th times. It is guaranteed that any three of the points given, the cities and the windmills, were not collinear, and that all of the windmills were on the same side of the line that the train ran along.

As shown in the picture, in Mr. Mian's perspective, B was initially to the left of A, and later to the right of A.

Input

The first line of input contains two integers n and m, where $n(1 \le n \le 1000)$ is number of windmills, and $m(1 \le m \le 10^4)$ is number of queries.

The second line contains four integers x_s, y_s, x_t and y_t $(-10^6 \le x_s, y_s, x_t, y_t \le 10^6)$, which are the coordinates of the starting city s and destination city t.

The next n lines describe the windmills, the i-th of which contains two integers x_i, y_i $(-10^6 \le x_i, y_i \le 10^6)$, which are the coordinates of the i-th windmill.

The next m lines describe the queries, the i-th of which contains two integers, h_i and k_i $(1 \le h_i \le n, 1 \le k_i \le 10^6)$, denoting a query for the coordinates when observing the k_i-th pass of the h_i-th windmill.

Output

Output m lines, each containing two real numbers x_i, y_i, representing the coordinates when the h_i-th windmill is observed to exchange order with other windmills for k times; if it does not exist, output -1. Your answer is considered correct if its absolute or relative error with the standard answer is less than 10^{-5}.

Example

standard input	standard output
4 2 0 0 5 0 1 3 2 4 4 1 4 5 1 2 3 2	-1 4.6666666667 0.0000000000

Problem K. Parallel Sort

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

As a master of parallel computing, schwer is recently considering about the method to achieve quick sorting on parallel computers. He needs your help!

Given a permutation $(p_1, \cdots, p_n)$, you need to sort the permutation with minimum number of **rounds**. In a single **round**, one can take many pairs of integers $(x_1, y_1), \cdots, (x_k, y_k)$ as long as the values of $x_1, y_1, \cdots, x_k, y_k$ are pairwise distinct. Then with the help of k CPUs, for each $i \in [1, k]$, the value of p_{x_i} and p_{y_i} will be switched immediately. Note that a permutation $(p_1, \cdots, p_n)$ is **sorted** if for every integer $i \in [1, n]$, $p_i = i$ holds.

Take some examples. Assume that $n = 4$. For $p = (1, 2, 3, 4)$, the minimum number of round is 0 as it is already sorted. For $p = (4, 3, 2, 1)$, the answer is 1, as you can swap the indices $(1, 4)$ and $(2, 3)$ simultaneously.

Input

The first line of input contains a single integer $n (1 \le n \le 10^5)$, indicating the length of the permutation.

The second line contains n integers $p_1, ..., p_n (1 \le p_i \le n)$, the given permutation. It is guaranteed that these integers form a permutation, that is, for every integer $i \in [1, n]$ there exists an unique integer $j \in [1, n]$ such that $p_j = i$.

Output

The first line of output should contain a single integer m, the minimum number of rounds to get the permutation sorted. Then print m line to show one possible solution.

The i-th of the next m lines should describe the i-th round of your solution, beginning with a single integer k, and followed by $2k$ integers $x_1, y_1; \cdots; x_k, y_k$. The constraints that $1 \le x_i, y_i \le n$ and the $2k$ integers are pairwise distinct must be held.

Examples

standard input	standard output
4 1 2 3 4	0
4 4 3 2 1	1 2 1 4 2 3

Problem L. Simone and Graph Coloring

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Simone, a student of Graph Coloring University, is interested in permutation. Now she is given a permutation of length n, and she finds that if she connects each inverse pair, she will get a graph. Formally, for the given permutation, if $i < j$ and $a_i > a_j$, then there will be an undirected edge between node i and node j in the graph.

Then she wants to color this graph. Please achieve poor Simone's dream. To simplify the problem, you just need to find a way of coloring the vertices of the graph such that no two adjacent vertices are of the same color and minimize the number of colors used.

Input

There are multiple test cases. The first line of the input contains an integer $T(1 \leq T \leq 10^6)$, indicating the number of test cases.

For each test case, the first line contains an integer $n(1 \leq n \leq 10^6)$, indicating the length of the permutation.

The second line contains n integers $a_1, a_2, ..., a_n$, indicating the permutation.

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

Output

For each test case, the first line contains an integer c, the chromatic number(the minimal number of colors been used when coloring) of the graph.

The second line contains n integers $c_1, c_2, ..., c_n$, the color of each node.

Notice that c_i should satisfy the limit that $1 \leq c_i \leq c$.

If there are several answers, it is acceptable to print any of them.

Example

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

Problem M. Stone Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

There are n piles of stones, the i-th of which contains s_i stones. The tiles are numbered from 1 to n. Rika and Satoko are playing a game on it.

During each round of the game, Rika chooses some piles from all n piles. Let denote the set of all chosen piles as S. Satoko then writes a non-negative integer x. If Rika takes some piles from S with the number of stones among all the taken piles equal to x, she wins the game, while otherwise (Rika cannot find such piles) Satoko wins the game. Note that each tile can be taken at most once during a round, and it is possible that Rika does not pick up any tile (when $x = 0$).

There are Q rounds of game in total, and for the i-th round of game Rika will let S be the set of all piles with index between l_i and r_i. For each round, Satoko wonders the minimum integer x she can write to win the game. As you are a master of programming, it is your turn to solve the problem!

Input

The first line of input contains two integers n and Q, where $n(1 \le n \le 10^6)$ is the number of tiles and $Q(1 \le Q \le 10^5)$ is the number of the rounds of the games.

The second line contains n integers $s_1, ..., s_n$, the i-th of which, $s_i(1 \le s_i \le 10^9)$, indicates the number of stones in the tile with index i.

Satoko wants you to compute the answers immediately after Rika chooses the interval, so she uses the following method to encrypt the input. The i-th of the next Q lines contains two integers $l'_i, r'_i(1 \le l'_i, r'_i \le n)$. The chosen index range for the i-th round, l_i, r_i, can be computed by the following formula:

$$l_i = \min\{(l'_i + ans_{i-1}) \bmod n + 1, (r'_i + ans_{i-1}) \bmod n + 1\}$$

$$r_i = \max\{(l'_i + ans_{i-1}) \bmod n + 1, (r'_i + ans_{i-1}) \bmod n + 1\}$$

where ans_i denotes the answer for the i-th round of the game (and thus ans_{i-1} is the answer for the previous round). You can assume that $ans_0 = 0$. It is clear that under the given constraints, $1 \le l_i \le r_i \le n$ holds.

Output

Output Q lines, the i-th of which contains a single integer ans_i, denoting the minimum integer x Satoko can choose to win the i-th round of the game.

Example

standard input	standard output
5 5 1 4 2 1 6 1 3 2 1 2 4 1 4 3 4	8 15 4 9 4

Note

In the example above, the actual query intervals are [2,4], [1,5], [3,5], [1,4] and [3,4].