2019-2020 ICPC, NERC, Southern and Volga Russian Regional Contest (Online Mirror, ICPC Rules, Teams Preferred)

A. Berstagram

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Polycarp recently signed up to a new social network Berstagram. He immediately published n posts there. He assigned numbers from 1 to n to all posts and published them one by one. So, just after publishing Polycarp's news feed contained posts from 1 to n — the highest post had number 1, the next one had number 2, ..., the lowest post had number n.

After that he wrote down all likes from his friends. Likes were coming consecutively from the 1-st one till the m-th one. You are given a sequence $a_1, a_2, \ldots, a_m$ ($1 \le a_j \le n$), where a_j is the post that received the j-th like.

News feed in Berstagram works in the following manner. Let's assume the j-th like was given to post a_j. If this post is not the highest (first) one then it changes its position with the one above. If a_j is the highest post nothing changes.

For example, if $n = 3$, $m = 5$ and $a = [3, 2, 1, 3, 3]$, then Polycarp's news feed had the following states:

- before the first like: $[1, 2, 3]$;
- after the first like: $[1, 3, 2]$;
- after the second like: $[1, 2, 3]$;
- after the third like: $[1, 2, 3]$;
- after the fourth like: $[1, 3, 2]$;
- after the fifth like: $[3, 1, 2]$.

Polycarp wants to know the highest (minimum) and the lowest (maximum) positions for each post. Polycarp considers all moments of time, including the moment "before all likes".

Input

The first line contains two integer numbers n and m ($1 \le n \le 10^5$, $1 \le m \le 4 \cdot 10^5$) — number of posts and number of likes.

The second line contains integers $a_1, a_2, \ldots, a_m$ ($1 \le a_j \le n$), where a_j is the post that received the j-th like.

Output

Print n pairs of integer numbers. The i-th line should contain the highest (minimum) and the lowest (maximum) positions of the i-th post. You should take into account positions at all moments of time: before all likes, after each like and after all likes. Positions are numbered from 1 (highest) to n (lowest).

Examples

input
3 5 3 2 1 3 3
output
1 2 2 3 1 3

input
10 6 7 3 5 7 3 6
output
1 2 2 3 1 3 4 7 4 5 6 7 5 7 8 8 9 9 10 10

B. The Feast and the Bus

time limit per test: 2 seconds

memory limit per test: 512 megabytes
input: standard input
output: standard output

Employees of JebTrains are on their way to celebrate the 256-th day of the year! There are n employees and k teams in JebTrains. Each employee is a member of some (exactly one) team. All teams are numbered from 1 to k. You are given an array of numbers $t_1, t_2, \ldots, t_n$ where t_i is the i-th employee's team number.

JebTrains is going to rent a single bus to get employees to the feast. The bus will take one or more rides. A bus can pick up an entire team or two entire teams. If three or more teams take a ride together they may start a new project which is considered unacceptable. It's prohibited to split a team, so all members of a team should take the same ride.

It is possible to rent a bus of any capacity s. Such a bus can take up to s people on a single ride. The total cost of the rent is equal to $s \cdot r$ burles where r is the number of rides. Note that it's impossible to rent two or more buses.

Help JebTrains to calculate the minimum cost of the rent, required to get all employees to the feast, fulfilling all the conditions above.

Input

The first line contains two integers n and k ($1 \le n \le 5 \cdot 10^5, 1 \le k \le 8000$) — the number of employees and the number of teams in JebTrains. The second line contains a sequence of integers $t_1, t_2, \ldots, t_n$, where t_i ($1 \le t_i \le k$) is the i-th employee's team number. Every team contains at least one employee.

Output

Print the minimum cost of the rent.

Examples

input
6 3 3 1 2 3 2 3
output
6

input
10 1 1 1 1 1 1 1 1 1 1 1
output
10

input
12 4 1 2 3 1 2 3 4 1 2 1 2 1
output
12

C. Trip to Saint Petersburg

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are planning your trip to Saint Petersburg. After doing some calculations, you estimated that you will have to spend k rubles each day you stay in Saint Petersburg — you have to rent a flat, to eat at some local cafe, et cetera. So, if the day of your arrival is L, and the day of your departure is R, you will have to spend $k(R - L + 1)$ rubles in Saint Petersburg.

You don't want to spend a lot of money on your trip, so you decided to work in Saint Petersburg during your trip. There are n available projects numbered from 1 to n, the i-th of them lasts from the day l_i to the day r_i inclusive. If you choose to participate in the i-th project, then you have to stay and work in Saint Petersburg for the entire time this project lasts, but you get paid p_i rubles for completing it.

Now you want to come up with an optimal trip plan: you have to choose the day of arrival L, the day of departure R and the set of projects S to participate in so that all the following conditions are met:

- your trip lasts at least one day (formally, $R \ge L$);
- you stay in Saint Petersburg for the duration of every project you have chosen (formally, for each $s \in S$ $L \le l_s$ and $R \ge r_s$);
- your total profit is **strictly positive** and maximum possible (formally, you have to maximize the value of $\sum\limits_{s \in S} p_s - k(R - L + 1)$, and this value should be positive).

You may assume that no matter how many projects you choose, you will still have time and ability to participate in all of them, even if they overlap.

Input
The first line contains two integers n and k ($1 \le n \le 2 \cdot 10^5$, $1 \le k \le 10^{12}$) — the number of projects and the amount of money you have to spend during each day in Saint Petersburg, respectively.

Then n lines follow, each containing three integers l_i, r_i, p_i ($1 \le l_i \le r_i \le 2 \cdot 10^5$, $1 \le p_i \le 10^{12}$) — the starting day of the i-th project, the ending day of the i-th project, and the amount of money you get paid if you choose to participate in it, respectively.

Output
If it is impossible to plan a trip with **strictly positive** profit, print the only integer 0.

Otherwise, print two lines. The first line should contain four integers p, L, R and m — the maximum profit you can get, the starting day of your trip, the ending day of your trip and the number of projects you choose to complete, respectively. The second line should contain m *distinct* integers s_1, s_2, ..., s_m — the projects you choose to complete, listed in arbitrary order. If there are multiple answers with maximum profit, print any of them.

Examples

input
4 5 1 1 3 3 3 11 5 5 17 7 7 4
output
13 3 5 2 3 2

input
1 3 1 2 5
output
0

input
4 8 1 5 16 2 4 9 3 3 24 1 5 13
output
22 1 5 4 3 2 1 4

D. Conference Problem

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

A large-scale conference on unnatural sciences is going to be held soon in Berland! In total, n scientists from all around the world have applied. All of them have indicated a time segment when they will attend the conference: two integers l_i, r_i — day of arrival and day of departure.

Also, some of the scientists have indicated their country, while some preferred not to. So, each scientist also has a value c_i, where:

- $c_i > 0$, if the scientist is coming from country c_i (all countries are numbered from 1 to 200);
- $c_i = 0$, if the scientist preferred to not indicate the country.

Everyone knows that it is interesting and useful to chat with representatives of other countries! A participant of the conference will be upset if she will not meet people from other countries during the stay. It is possible to meet people during all time of stay, including the day of arrival and the day of departure.

Conference organizers need to be ready for the worst! They are interested in the largest number x, that it is possible that among all people attending the conference exactly x will be upset.

Help the organizers to find the maximum number of upset scientists.

Input
The first line of the input contains integer t ($1 \le t \le 100$) — number of test cases. Then the test cases follow.

The first line of each test case contains integer n ($1 \le n \le 500$) — the number of registered conference participants.

Next n lines follow, each containing three integers l_i, r_i, c_i ($1 \le l_i \le r_i \le 10^6$, $0 \le c_i \le 200$) — the day of arrival, the day of departure and the country (or the fact that it was not indicated) for the i-th participant.

The sum of n among all test cases in the input does not exceed 500.

Output
Output t integers — maximum number of upset scientists for each test case.

Example

input
2 4 1 10 30 5 6 30 6 12 0 1 1 0 4 1 2 1 2 3 0 3 4 0 4 5 2
output
4 2

E. The Coronation

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

The coronation of King Berl XXII is soon! The whole royal family, including n daughters of Berl XXII, will be present.

The King has ordered his jeweler to assemble n beautiful necklaces, so each of the princesses could wear exactly one necklace during the ceremony — and now these necklaces are finished. Each necklace consists of m gems attached to a gold chain. There are two types of gems used in the necklaces — emeralds and sapphires. So, each necklace can be represented by a sequence of m gems (listed from left to right), and each gem is either an emerald or a sapphire. Formally, the i-th necklace can be represented by a binary string s_i of length m; if the j-th character of s_i is `0`, then the j-th gem in the i-th necklace is an emerald; otherwise, this gem is a sapphire.

Now, looking at the necklaces, the King is afraid that some of his daughters may envy the other daughters' necklaces. He wants all necklaces to look *similar*. Two necklaces are considered *similar* if there are at least k positions where these necklaces contain the same type of gems.

For example, if there is a necklace represented by a sequence `01010111` and a necklace represented by a sequence `01100000`, then there are 3 positions where these necklaces contain the same type of gems (both first gems are emeralds, both second gems are sapphires, and both fifth gems are emeralds). So if $k = 3$, these necklaces are *similar*, and if $k = 4$, they are not *similar*.

The King thinks that if two of his daughters notice that their necklaces are not *similar*, then they may have a conflict — and, obviously, he doesn't want any conflicts during the coronation! So Berl XXII wants to tell some of his daughters to wear their necklaces backward. If a necklace is worn backward, then the sequence of gems in this necklace is reversed. For example, if a necklace is represented by a sequence `01100`, then, if worn backward, it would be represented by a sequence `00110`. The King wants to find the minimum number of necklaces to be worn backward during the coronation so that there are no conflicts.

Berl XXII is too busy with preparation for the coronation, so he ordered you to resolve this issue for him. Help him — and he will give you a truly royal reward!

Input
The first line contains one integer t ($1 \le t \le 50$) — the number of test cases. Then the test cases follow.

Each test case begins with a line containing three integers n, m and k ($2 \le n \le 50$, $1 \le k \le m \le 50$) — the number of necklaces, the number of gems in each necklace, and the minimum number of positions where two necklaces have to have the same type of gems in order to look similar, respectively.

Then n lines follow, the i-th of them contains a binary string s_i of length m representing the i-th necklace.

Output
For each test case, print the answer as follows.

If it is impossible to avoid the conflict, print `-1` on a single line. In this case you should not output anything else for that test case.

Otherwise, the first line of the test case answer should contain the single integer d — the minimum number of necklaces that are to be worn backward. The second line of the test case answer should contain the numbers of these necklaces (integers from 1 to n) in any order. If $d = 0$ then leave the second line of the test case answer empty. If there are multiple answers, you may print any of them.

Example

input
5 5 7 2

```
1010100
0010101
1111010
1000010
0000101
6 9 3
011111110
100111000
111100000
000111111
110100111
111110111
3 4 2
0001
1000
0000
3 4 4
0001
1000
0000
2 4 3
0001
1000
```

output

```
2
1 3
1
3
0

-1
1
1
```

F. Data Center

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are developing a project to build a new data center. The data center will be a rectangle with an area of exactly n square meters. Each side of the data center must be an integer.

Your goal is to minimize the impact of the external environment on the data center. For this reason, you want to minimize the length of the perimeter of the data center (that is, the sum of the lengths of its four sides).

What is the minimum perimeter of a rectangular data center with an area of exactly n square meters, if the lengths of all its sides must be integers?

Input

The first and only line of the input contains an integer n ($1 \le n \le 10^5$), where n is the area of the data center in square meters.

Output

Print the required minimum perimeter in meters.

Examples

input
36
output
24

input
13
output
28

input
1
output
4

Note

In the first example, the required shape of the data center is 6×6 square. Its area is 36 and the perimeter is $6 + 6 + 6 + 6 = 24$.

In the second example, the required shape of the data center is 1×13 rectangle. Its area is 13 and the perimeter is $1 + 13 + 1 + 13 = 28$.

In the third example, the required shape of the data center is 1×1 square. Its area is 1 and the perimeter is $1 + 1 + 1 + 1 = 4$.

G. Discarding Game

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Eulampius has created a game with the following rules:

- there are two players in the game: a human and a computer;
- the game lasts for no more than n rounds. Initially both players have 0 points. In the j-th round the human gains a_j points, and the computer gains b_j points. The points are gained simultaneously;
- the game ends when one of the players gets k or more points. This player loses the game. If both players get k or more points simultaneously, both lose;
- if both players have less than k points after n rounds, the game ends in a tie;
- after each round the human can push the "Reset" button. If the human had x points, and the computer had y points before the button is pushed (of course, $x < k$ and $y < k$), then after the button is pushed the human will have $x' = max(0,\, x - y)$ points, and the computer will have $y' = max(0,\, y - x)$ points. E. g. the push of "Reset" button transforms the state $(x = 3,\, y = 5)$ into the state $(x' = 0,\, y' = 2)$, and the state $(x = 8,\, y = 2)$ into the state $(x' = 6,\, y' = 0)$.

Eulampius asked his friend Polycarpus to test the game. Polycarpus has quickly revealed that amounts of points gained by the human and the computer in each of n rounds are generated before the game and stored in a file. In other words, the pushes of the "Reset" button do not influence the values a_j and b_j, so sequences a and b are fixed and known in advance.

Polycarpus wants to make a plan for the game. He would like to win the game pushing the "Reset" button as few times as possible. Your task is to determine this minimal number of pushes or determine that Polycarpus cannot win.

Input

The first line of the input contains one integer t ($1 \le t \le 10000$) — the number of test cases. Then the test cases follow.

The first line of each test case contains two integers n and k ($1 \le n \le 2 \cdot 10^5$, $2 \le k \le 10^9$) — the maximum possible number of rounds in the game and the number of points, after reaching which a player loses, respectively.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_j < k$), where a_j is the amount of points the human gains in the j-th round.

The third line of each test case contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_j < k$), where b_j is the amount of points the computer gains in the j-th round.

The sum of n over all test cases in the input does not exceed $2 \cdot 10^5$.

Output

Print the answers for all test cases in the order they appear in the input.

If Polycarpus cannot win the game, then simply print one line "-1" (without quotes). In this case, you should not output anything else for that test case. Otherwise, the first line of the test case answer should contain one integer d — the minimum possible number of "Reset" button pushes, required to win the game. The next line should contain d distinct integers $r_1, r_2, \ldots, r_d$ ($1 \le r_i < n$) — the numbers of rounds, at the end of which Polycarpus has to press the "Reset" button, in arbitrary order. If $d = 0$ then either leave the second line of the test case answer empty, or do not print the second line at all.

If there are several possible solutions, print any of them.

Example

input
3 4 17 1 3 5 7 3 5 7 9 11 17 5 2 8 2 4 6 1 2 7 2 5 4 6 3 3 5 1 7 4 2 5 3 6 17 6 1 2 7 2 5 1 7 4 2 5 3
output
0 2 2 4 -1

Note

In the second test case, if the human pushes the "Reset" button after the second and the fourth rounds, the game goes as follows:

1. after the first round the human has 5 points, the computer — 4 points;

2. after the second round the human has 7 points, the computer — 10 points;
3. the human pushes the "Reset" button and now he has 0 points and the computer — 3 points;
4. after the third round the human has 8 points, the computer — 6 points;
5. after the fourth round the human has 10 points, the computer — 9 points;
6. the human pushes "Reset" button again, after it he has 1 point, the computer — 0 points;
7. after the fifth round the human has 5 points, the computer — 5 points;
8. after the sixth round the human has 11 points, the computer — 6 points;
9. after the seventh round the human has 12 points, the computer — 13 points;
10. after the eighth round the human has 14 points, the computer — 17 points;
11. the human wins, as the computer has k or more points and the human — strictly less than k points.

H. Happy Birthday

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You have a set of birthday cake candles. Each of such candles represents a digit between 0 and 9, inclusive.

Example of birthday cake candles.

Let's denote the candle representing the digit d as d-candle.

Your set contains c_0 instances of 0-candles, c_1 instances of 1-candles and so on. So, the total number of candles is $c_0 + c_1 + \cdots + c_9$.

These digits are needed to wish your cat a happy birthday. For each birthday, starting with the first, you want to compose the age of the cat using the digits from the set.

Since you light candles for a very short time, candles don't have time to burn out. For this reason *you can reuse candles an arbitrary number of times* (therefore your set of candles never changes).

For example, if you have one instance of each digit (i.e. $c_0 = c_1 = \cdots = c_9 = 1$), you can compose any number from 1 to 10 using this set, but you cannot compose 11.

You have to determine the first birthday, on which you cannot compose the age of the cat using the candles from your set. In other words, find the minimum number y such that all numbers from 1 to $y - 1$ can be composed by digits from your set, but y cannot be composed.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases in the input.

The only line of each test case contains ten integer numbers $c_0, c_1, \ldots, c_9$ ($0 \le c_i \le 10^5$) — the number of 0-candles, 1-candles, 2-candles and so on.

It is guaranteed that the sum of all c_i in the input does not exceed 10^6.

Output

For each test case, output one integer in single line — the minimum age which cannot be composed by candles from your set. Please note that the age can be quite large (it may exceed the standard 64-bit integer types in your programming language).

Example

input
4 1 1 1 1 1 1 1 1 1 1 0 0 1 1 2 2 3 3 4 4 1 2 1 2 1 3 1 0 0 0 0 1 2 1 4 3 1 1 2 1
output
11 1 7 10

I. Show Must Go On

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

The director of the famous dance show plans a tour. It is already decided that the tour will consist of up to m concerts.

There are n dancers in the troupe. Each dancer is characterized by her awkwardness: the awkwardness of the i-th dancer is equal to a_i.

The director likes diversity. For this reason, each concert will be performed by a different set of dancers. A dancer may perform in multiple concerts. For example, it is possible that a set of dancers performs in one concert and a subset of this set of dancers performs in another concert. The only constraint is that the same set of dancers cannot perform twice.

The director prefers the set with larger number of dancers over the set with smaller number of dancers. If two sets consist of the same number of dancers, then the director prefers the one which has smaller sum of awkwardness of dancers. If two sets of dancers are equal in size and total awkwardness, then the director does not have a preference which one is better.

A marketing study shows that viewers are not ready to come to a concert if the total awkwardness of all the dancers performing in the concert is greater than k.

The director wants to find the best plan for m concerts. He thinks to write down all possible sets of dancers; then get rid of the sets with total awkwardness greater than k. The remaining sets of dancers will be sorted according to his preference. The most preferred set of dancers will give the first concert, the second preferred set — the second concert and so on until the m-th concert. If it turns out that the total number of valid sets is less than m, then the total number of concerts will be equal to the number of valid sets.

It turns out that the director delegated finding the plan to you! Please, notice that there might be several acceptable plans due to the fact that the director does not have a preference over sets of dancers with the same size and total awkwardness. In this case any of these plans is good enough. For each concert find the number of dancers and the total awkwardness of the set performing. Also, for the last concert find its set of dancers.

Input

The first line contains one integer t ($1 \le t \le 10^5$) — the number of test cases in the input. Then the test cases follow.

Each test case begins with a line containing three integers n, k and m ($1 \le n \le 10^6$, $1 \le k \le 10^{18}$, $1 \le m \le 10^6$) — the total number of dancers, the maximum acceptable awkwardness of a set of dancers and the maximum number of concerts, respectively.

The following line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{12}$), where a_i is the awkwardness of the i-th dancer.

The sum of the values of n over all test cases in the input does not exceed 10^6. Similarly, the sum of the values of m over all test cases in the input does not exceed 10^6.

Output

Print the answers to all test cases in the input.

If the troupe cannot give concerts at all, then simply print one line "`0`". In this case, you should not print anything else.

If the troupe gives a positive number of concerts r (r is equal to the minimum of m and the total number of valid sets), then first print the value of r, then r lines: the j-th line should contain two integers s_j and t_j — the number of dancers in the j-th concert and the total awkwardness of the dancers performing in the j-th concert. Complete the output to a test case with a line that describes the last set: print exactly s_r distinct integers from 1 to n — the numbers of the dancers who will perform at the r-th (last) concert, in any order. If there are several answers, print any of them.

Example

input
3 7 13 10 3 1 5 1 8 2 13 2 10 1 12 12 3 32 100000 2 1 5
output
10 5 12 4 7 4 9 4 10 4 11 4 11 4 12 4 13 3 4 3 5 2 4 1 0 7 3 8

```
2 3
2 6
2 7
1 1
1 2
1 5
3
```

J. The Parade

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

The Berland Army is preparing for a large military parade. It is already decided that the soldiers participating in it will be divided into k rows, and all rows will contain *the same* number of soldiers.

Of course, not every arrangement of soldiers into k rows is suitable. Heights of all soldiers in the same row should not differ by more than 1. The height of each soldier is an integer between 1 and n.

For each possible height, you know the number of soldiers having this height. To conduct a parade, you have to choose the soldiers participating in it, and then arrange *all of the chosen soldiers* into k rows so that both of the following conditions are met:

- each row has the same number of soldiers,
- no row contains a pair of soldiers such that their heights differ by 2 or more.

Calculate the maximum number of soldiers who can participate in the parade.

Input
The first line contains one integer t ($1 \le t \le 10000$) — the number of test cases. Then the test cases follow.

Each test case begins with a line containing two integers n and k ($1 \le n \le 30000$, $1 \le k \le 10^{12}$) — the number of different heights of soldiers and the number of rows of soldiers in the parade, respectively.

The second (and final) line of each test case contains n integers c_1, c_2, ..., c_n ($0 \le c_i \le 10^{12}$), where c_i is the number of soldiers having height i in the Berland Army.

It is guaranteed that the sum of n over all test cases does not exceed 30000.

Output
For each test case, print one integer — the maximum number of soldiers that can participate in the parade.

Example

input
5 3 4 7 1 13 1 1 100 1 3 100 2 1 1000000000000 1000000000000 4 1 10 2 11 1
output
16 100 99 2000000000000 13

Note
Explanations for the example test cases:

1. the heights of soldiers in the rows can be: $[3, 3, 3, 3]$, $[1, 2, 1, 1]$, $[1, 1, 1, 1]$, $[3, 3, 3, 3]$ (each list represents a row);
2. all soldiers can march in the same row;
3. 33 soldiers with height 1 in each of 3 rows;
4. all soldiers can march in the same row;
5. all soldiers with height 2 and 3 can march in the same row.

K. Projectors

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

There are n lectures and m seminars to be conducted today at the Faculty of Approximate Sciences. The i-th lecture starts at a_i and ends at b_i (formally, time of the lecture spans an interval $[a_i, b_i)$, the right bound is exclusive). The j-th seminar starts at p_j and ends at q_j (similarly, time of the seminar spans an interval $[p_j, q_j)$, the right bound is exclusive).

There are x HD-projectors numbered from 1 to x and y ordinary projectors numbered from $x + 1$ to $x + y$ available at the faculty. Projectors should be distributed in such a way that:

- an HD-projector is used in each lecture;
- some projector (ordinary or HD) is used in each seminar;
- a projector (ordinary or HD) can only be used in one event at the same moment of time;
- if a projector is selected for an event, it is used there for the whole duration of the event;
- a projector can be reused in some following event, if it starts not earlier than current event finishes.

You are to find such distribution of projectors, if it exists.

Again, note that the right bound of the event's time range is not inclusive: if some event starts exactly when another event finishes, the projector can be reused (suppose that it is instantly transported to the location of the event).

Input
The first line contains an integer t ($1 \le t \le 300$) — the number of test cases.

Each test case starts with a line containing four integers n, m, x, y ($0 \le n, m, x, y \le 300$; $n + m > 0$, $x + y > 0$) — the number of lectures, the number of seminars, the number of HD projectors and the number of ordinary projectors, respectively.

The next n lines describe lectures. Each line contains two integers a_i, b_i ($1 \le a_i < b_i \le 10^6$) — the start time (inclusive) and finish time (exclusive) of the i-th lecture.

The next m lines describe seminars. Each line contains two integers p_j, q_j ($1 \le p_j < q_j \le 10^6$) — the start time (inclusive) and finish time (exclusive) of the j-th seminar.

Output
For each test case, print YES if it is possible to distribute projectors in order to meet all requirements, or NO otherwise.

In case of positive answer, output one additional line containing $n + m$ integers. The first n integers should be not less than 1 and not greater than x, and the i-th of them should be the index of HD projector used in the i-th lecture. The last m integers should be not less than 1 and not greater than $x + y$, and the j-th of them should be the index of projector used in the j-th seminar. If there are multiple answers, print any of them.

Examples

input
2 2 2 2 2 1 5 2 5 1 5 1 4 2 0 2 10 1 3 1 3
output
YES 2 1 4 3 YES 2 1

input
3 1 2 1 1 3 4 2 4 1 3 3 4 2 3 5 7 1 3 1 7 4 8 2 5 1 6 2 8 0 1 1 0 1 1000000
output
YES 1 2 1 NO YES 1

L. Divide The Students

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

A group of students has recently been admitted to the Faculty of Computer Sciences at the Berland State University. Now the programming teacher wants to divide them into three subgroups for practice sessions.

The teacher knows that a lot of programmers argue which language is the best. The teacher doesn't want to hear any arguments in the subgroups, so she wants to divide the students into three subgroups so that no pair of students belonging to the same subgroup want to argue.

To perform this division, the teacher asked each student which programming language he likes. There are a students who answered that they enjoy Assembler, b students stated that their favourite language is Basic, and c remaining students claimed that C++ is the best programming language — and there was a large argument between Assembler fans and C++ fans.

Now, knowing that Assembler programmers and C++ programmers can start an argument every minute, the teacher wants to divide the students into three subgroups so that every student belongs to exactly one subgroup, and there is no subgroup that contains at least one Assembler fan and at least one C++ fan. Since teaching a lot of students can be difficult, the teacher wants the size of the largest subgroup to be minimum possible.

Please help the teacher to calculate the minimum possible size of the largest subgroup!

Input

The first line contains one integer t ($1 \le t \le 5$) — the number of test cases in the input. Then test cases follow.

Each test case consists of one line containing three integers a, b and c ($1 \le a, b, c \le 1000$) — the number of Assembler fans, Basic fans and C++ fans, respectively.

Output

For each test case print one integer — the minimum size of the largest subgroup if the students are divided in such a way that there is no subgroup that contains at least one Assembler fan and at least one C++ fan simultaneously.

Examples

input
5 3 5 7 4 8 4 13 10 13 1000 1000 1000 13 22 7

output
5 6 13 1000 14

input
5 1 3 4 1000 1000 1 4 1 2 325 226 999 939 861 505

output
3 667 3 517 769

Note

Explanation of the answers for the example 1:

1. The first subgroup contains 3 Assembler fans and 2 Basic fans, the second subgroup — 5 C++ fans, the third subgroup — 2 C++ fans and 3 Basic fans.
2. The first subgroup contains 4 Assembler fans, the second subgroup — 6 Basic fans, the third subgroup — 2 Basic fans and 4 C++ fans.
3. The first subgroup contains all Assembler fans, the second subgroup — all Basic fans, the third subgroup — all C++ fans.
4. The first subgroup contains all Assembler fans, the second subgroup — all Basic fans, the third subgroup — all C++ fans.
5. The first subgroup contains 12 Assembler fans and 2 Basic fans, the second subgroup — 1 Assembler fan and 13 Basic fans, the third subgroup — 7 Basic fans and 7 C++ fans.

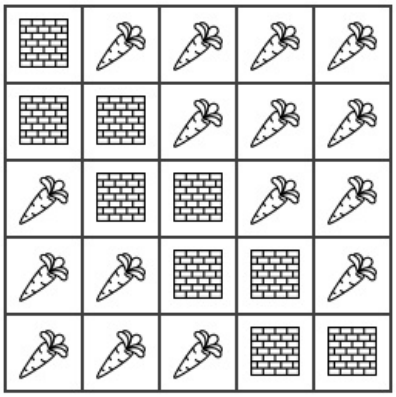
M. SmartGarden

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Berland Gardeners United Inc. hired you for the project called "SmartGarden". The main feature of this project is automatic garden watering.

Formally the garden can be represented as a square of $n \times n$ cells with rows numbered 1 to n from top to bottom and columns numbered 1 to n from left to right. Each cell of the garden contains either a plant or a slab.

It's known that slabs are located on the main diagonal of the matrix representing the garden, and in the cells that are below the main diagonal and share a side with at least one cell of the main diagonal. All the remaining cells of the garden are filled with plants.



Example of the garden for $n = 5$.

During implementation of the project you created a smart robot that takes a list of commands as an input, which are processed one by one. Each command contains:

- a list of horizontal lines (rows in the matrix representing the garden);
- a list of vertical lines (columns in the matrix representing the garden).

While executing each command robot waters only cells in the intersection of specified rows and specified columns. So, if you specify r rows and c columns, then exactly $r \cdot c$ cells will be watered.

In the demo for the customer you have tuned robot in such a way that it waters all the garden. To do that you prepared a single command containing all n rows and all n columns.

Unfortunately, 5 hours before the demo for your customer it turned out that the CEO of Berland Gardeners United Inc. was going to take part in it. Moreover, most probably he will be standing on a garden slab during the demo!

Now you need to create a list of commands for the robot so that it waters all the plants and doesn't water any cell containing a slab. Since it's only a beta version of "SmartGarden", the total number of commands shouldn't exceed 50.

Create a program that, for a given size of the garden, will find a list of no more than 50 commands that allow the robot to water all the plants in the garden without watering the slabs. It is allowed to water a plant several times.

Input

The first and the only line of the input contains a single integer n ($2 \le n \le 5000$), where n is the size of the garden.

Output

In the first line print the total number of commands for the robot k ($1 \le k \le 50$). In the next $2 \cdot k$ lines print all the commands. Each command should be specified by 2 lines. The first line of each command should describe rows in the command and the second line should describe columns in the command. Each of these 2 lines should have the following format:

- the first number of the line should specify the total number of items x in the appropriate list;
- then x **distinct** numbers follow, where each number is in the range $1 \dots n$ and describes a chosen row for the first line and a chosen column for the second line.

If there are multiple ways to water the garden, print any of them.

Examples

input
2

output
2 1 1 1 2 1 1 1 2

input

```
4
```

output

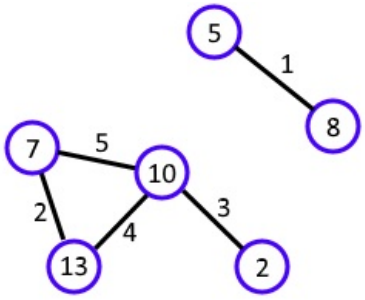
```
4
2 1 4
1 2
2 1 2
2 3 4
1 3
2 1 4
1 4
1 1
```

N. Wires

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Polycarpus has a complex electronic device. The core of this device is a circuit board. The board has 10^9 contact points which are numbered from 1 to 10^9. Also there are n wires numbered from 1 to n, each connecting two distinct contact points on the board. An electric signal can pass between wires A and B if:

- either both wires share the same contact point;
- or there is a sequence of wires starting with A and ending with B, and each pair of adjacent wires in the sequence share a contact point.



The picture shows a circuit board with 5 wires. Contact points with numbers $2, 5, 7, 8, 10, 13$ are used. Here an electrical signal can pass from wire 2 to wire 3, but not to wire 1.

Currently the circuit board is broken. Polycarpus thinks that the board could be fixed if the wires were re-soldered so that a signal could pass between any pair of wires.

It takes 1 minute for Polycarpus to re-solder an end of a wire. I.e. it takes one minute to change one of the two contact points for a wire. Any contact point from range $[1, 10^9]$ can be used as a new contact point. A wire's ends must always be soldered to distinct contact points. Both wire's ends can be re-solded, but that will require two actions and will take 2 minutes in total.

Find the minimum amount of time Polycarpus needs to re-solder wires so that a signal can pass between any pair of wires. Also output an optimal sequence of wire re-soldering.

Input

The input contains one or several test cases. The first input line contains a single integer t — number of test cases. Then, t test cases follow.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$) — the number of wires. The following n lines describe wires, each line containing two space-separated integers x_i, y_i ($1 \le x_i, y_i \le 10^9$, $x_i \ne y_i$) — contact points connected by the i-th wire. A couple of contact points can be connected with more than one wire.

Sum of values of n across all test cases does not exceed 10^5.

Output

For each test case first print one line with a single integer k — the minimum number of minutes needed to re-solder wires so that a signal can pass between any pair of wires. In the following k lines print the description of re-solderings. Each re-soldering should be described by three integers w_j, a_j, b_j ($1 \le w_j \le n$, $1 \le a_j, b_j \le 10^9$). Such triple means that during the j-th re-soldering an end of the w_j-th wire, which was soldered to contact point a_j, becomes soldered to contact point b_j instead. After each re-soldering of a wire it must connect two distinct contact points. If there are multiple optimal re-solderings, print any of them.

Example

input

```
2
1
4 7
4
1 2
2 3
4 5
```

5 6

output

0
1
2 3 5