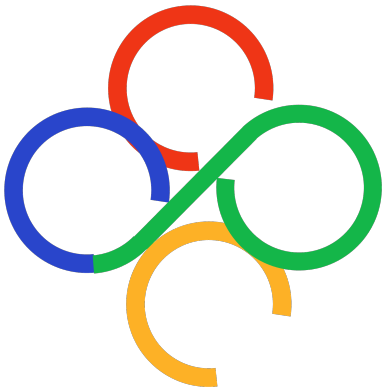
2022 China Collegiate Programming Contest

Weihai Site



哈爾濱工業大學（威海）

Harbin Institute of Technology, Weihai

Problems

Problem A. Dunai

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

Lope the Bear often watches game events. He has a special skill: Dunai, also known as poisoned milk. Every time he predicts the outcome of a game by wishing the team that he believes will win good luck, the outcome often turns out to be the opposite. Recently, a tournament called *Daota 2 The International* is about to start. It's time for Lope to show his Dunai skill again.

In the game *Daota 2*, each team has five players corresponding to five positions, numbered from 1 to 5. *Daota 2 The International* has been held for n editions. To better Dunai, Lope looks up the champion team of each edition, including the names and positions of the five players. In addition, he inquires about the m players involved in the upcoming tournament including their names and positions, without knowing exactly how the teams will be formed. What is known is that for every player who has won a championship, his position in the team never changes.

Lope wants to predict how these m players will be teamed up in a way that satisfies the following conditions:

- Each player is assigned to at most one team (possibly not assigned).
- The team consists of five different positions, with exactly one player for each position.
- Each team includes at least one player who has won a championship.

Lope wants to know the maximum number of teams that can be formed.

Input

The first line contains an integer n ($1 \le n \le 100$), indicating the number of editions that the tournament has been held for.

Each of the following n lines contains five strings separated by spaces, indicating the name of five players in the champion team, corresponding to positions 1 to 5 respectively.

The next line contains an integer m ($1 \le m \le 1000$), indicating the number of players involved in the upcoming tournament.

Each of the following m lines contains a string and an integer between 1 and 5, indicating the name and position of a player. It is guaranteed that the names of the players are all distinct, and for every player who has won a championship, his position in the team never changes.

It is guaranteed that the string representing each player's name consists of only English letters and digital numbers, and the length of the string does not exceed 20.

Output

Output an integer indicating the maximum number of teams that can be formed.

Example

standard input	standard output
1 Kanon Keke Chisato Sumire Ren 9 Kanon 1 Keke 2 Chisato 3 Sumire 4 Ren 5 Kinako 1 Mei 2 Shiki 3 Natsumi 4	1

Note

Please refer to the online judge for the second sample.

Problem B. Recruitment

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

Cody the Wolf is the coach of a programming competition team. With the new contest season approaching, Cody intends to recruit some new members to the team. He prepares a programming problem for this recruitment:

- There is an expression with n positive integers and $n-1$ plus signs $a_1 + a_2 + \cdots + a_n$. We replace one plus sign $+$ with a multiplication sign $\times$ each time and perform $n-1$ times in total. Please calculate the result of the expression after each replacement.

Cody has generated the standard input and output files for this problem. But he deletes all the standard input files by mistake. Overwhelmed, Cody wonders if he can regenerate the corresponding input files while leaving the output files unchanged.

Formally, given n integers $s_0, s_1, \ldots, s_{n-1}$, where s_i indicates the result of the expression after the i-th replacement, you need to construct an initial expression $a_1 + a_2 + \cdots + a_n$ and determine the position of the plus sign for each replacement such that the result of each replacement matches the given integers $s_0, s_1, \ldots, s_{n-1}$.

Input

The first line contains an integer n ($1 \le n \le 10^5$), indicating the number of integers in the expression.

The second line contains n integers $s_0, s_1, \ldots, s_{n-1}$ ($1 \le s_i \le 10^9$), where s_i indicates the result of the expression after the i-th replacement.

Output

If there is no possible solution, output -1 in a single line.

Otherwise, output n positive integers $a_1, a_2, \ldots, a_n$ in the first line separated by spaces, indicating that the initial expression is $a_1 + a_2 + \cdots + a_n$. Then output $n-1$ lines, the i-th of which contains an integer indicating the position of the plus sign for the i-th replacement. You need to make the $n-1$ integers a permutation of 1 to $n-1$, i.e., each integer from 1 to $n-1$ occurs exactly once.

If there are multiple solutions, output any.

Examples

standard input	standard output
4 13 12 19 60	5 3 4 1 3 1 2
10 10 9 8 7 6 5 4 3 2 1	1 1 1 1 1 1 1 1 1 1 1 2 3 4 5 6 7 8 9
6 1 1 4 5 1 4	-1

Note

For the first sample:

- The expression after the 0th replacement i.e. the initial expression is $5 + 3 + 4 + 1 = 13$.
- The expression after the 1st replacement is $5 + 3 + 4 \times 1 = 12$.
- The expression after the 2nd replacement is $5 \times 3 + 4 \times 1 = 19$.
- The expression after the 3rd replacement is $5 \times 3 \times 4 \times 1 = 60$.

Problem C. Grass

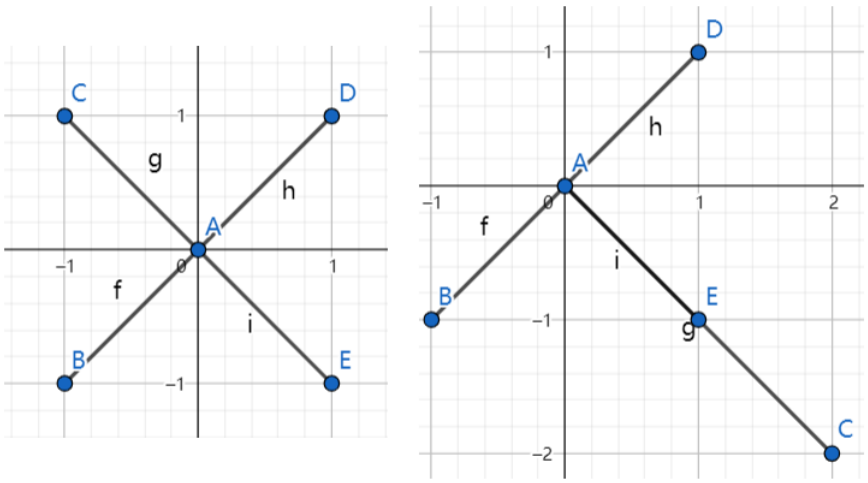
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Charles the Rabbit likes eating grass. As the saying goes, rabbits do not eat the grass by their burrows. Therefore, Charles has to go outside his burrow every day to look for grass to eat.

One day, Charles comes to a two-dimensional plane with many distinct points. He can choose a point A and another four points B, C, D, E to connect with A to form four segments. We consider these four segments as a clump of grass if they meet the following condition:

- Any two of the four segments have only a single point of intersection A between them.

For example, in the picture below, (1) is a clump of grass, but (2) is not one as the intersection of segments AC and AE is not only a single point A.



(1) (2)

Given n points on a plane, Charles wants to know whether there exists a clump of grass. If so, help him find a certain one.

Input

The first line contains an integer T ($1 \le T \le 120$), indicating the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 25000$), indicating the number of points.

Each of the following n lines contains two integers x, y ($-10^7 \le x, y \le 10^7$), indicating that the coordinates of the point are (x, y). It is guaranteed that all points are distinct.

It is guaranteed that $\sum n \le 10^5$ over all test cases.

Output

For each test case, if there does not exist a clump of grass, output `NO` in a single line.

Otherwise, output `YES` in the first line. Then output two integers separated by a space in the second line, indicating the coordinates of point A. Then output two integers separated by a space in each of the third to sixth lines, indicating the coordinates of the other four points B, C, D, E.

If there is more than one clump of grass, output any.

Example

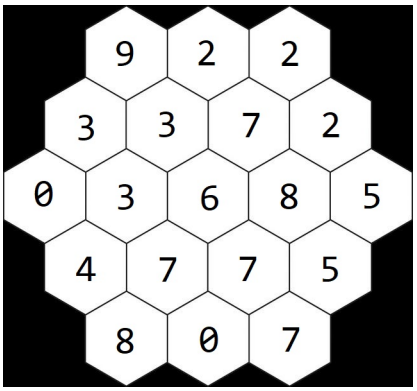
standard input	standard output
3 5 0 0 1 1 1 -1 -1 1 -1 -1 3 1 1 4 5 1 4 5 1 0 2 0 3 0 4 0 5 0	YES 0 0 1 1 1 -1 -1 1 -1 -1 NO NO

Problem D. Sternhalma

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

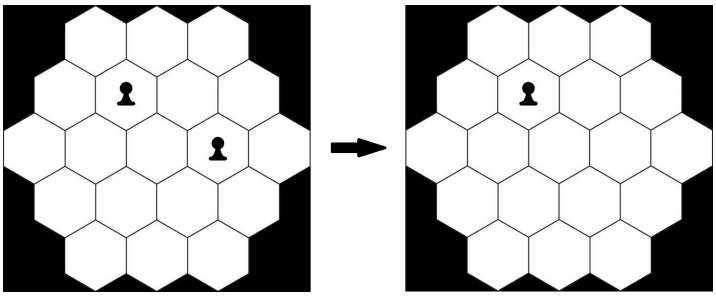
Pico the Puppy, together with FuuFuu and Kotsuki, lives in the KFP apartment. They often play a board game together called Sternhalma, commonly known as Chinese checkers. The objective of the game is to race all of one's pieces across the hexagram-shaped board to the opposite side, using single-step moves or moves that jump over other pieces.

One day, Pico wants to play Sternhalma again, but Kotsuki is out at work, and FuuFuu is still sleeping. Feeling bored, Pico intends to play on his own for a while. Pico simplifies the board into the shape shown in the picture below, with a total of 19 grids, and he assigns a score to each grid.

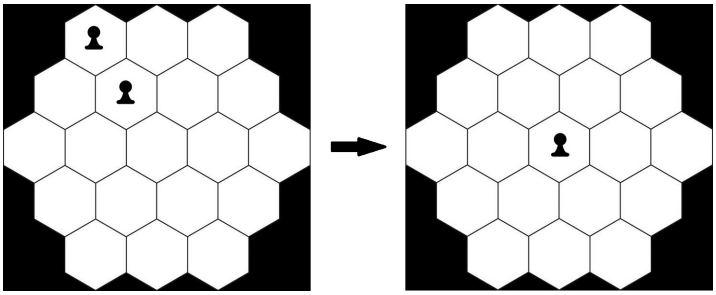


Initially, he places a number of pieces on the board. Then, he moves the pieces according to the rule set by himself, selecting either of the following two types of moves for each turn:

- Remove any piece directly from the board, getting no scores.



- Remove a piece by jumping over it. Formally, for two adjacent pieces A and B, if the symmetric position of A with respect to B is not out of bounds and no piece is placed there, then A can jump over B, and B is removed from the board. The total score will be increased by the score assigned to the grid where B is located before being removed. (We consider two pieces adjacent if and only if the grids where the two pieces are located share an edge.)



The initial score is 0. Pico will keep removing the pieces until there are no pieces left on the board. For different initial placements of pieces, he wants to know the maximum score he can get.

Input

The first five lines contain 19 integers, indicating the scores assigned to the grids. The first line contains 3 integers indicating the first row of the board, the second line contains 4 integers indicating the second row of the board, and so on. Each score ranges between -10^6 and 10^6.

The next line contains an integer n ($1 \le n \le 10^4$), indicating the number of initial placements of pieces.

Each of the n initial placements occupies five lines. Each line contains a string consisting of only `.` or `#`. The first line contains 3 characters indicating the first row of the board, the second line contains 4 characters indicating the second row of the board, and so on. `#` denotes that there is a piece at this position, while `.` denotes that there is none.

Output

For each initial placement of pieces, output the maximum score in a single line.

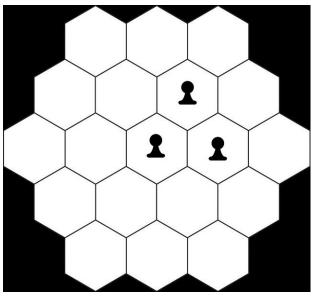
Example

standard input	standard output
9 2 2 3 3 7 2 0 3 6 8 5 4 7 7 5 8 0 7 3 ... ..#. ..##. ....##.. ..#. ... ### #### ##### #### ###	8 14 105

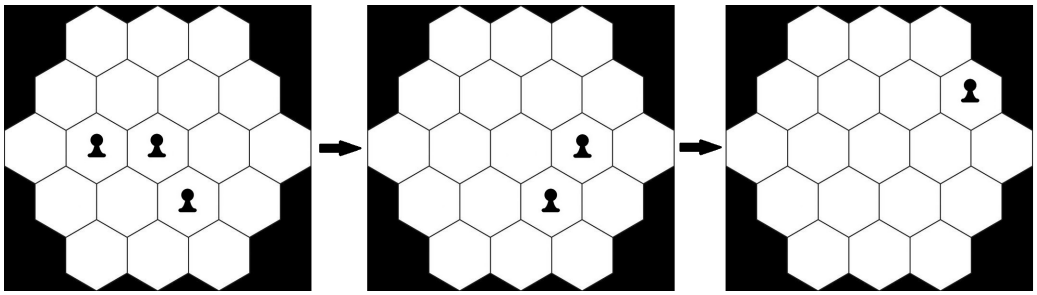
Note

The first initial placement of the sample is shown in the picture below. Obviously, only one piece can be

removed by jumping over it.

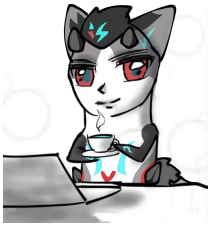


For the second initial placement of the sample, the moves to obtain the maximum score are shown in the following picture.



Problem E. Python Will be Faster than C++

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



Little Q is learning programming languages. He often uses Python as it is one of the most used programming languages. However, when he codes in Python, he finds that the programs do not run very efficiently, especially when compared to C/C++. He wonders if the running efficiency of Python has any improvement with version updates, and conducts an experiment.

The algorithm used in the experiment is Monte Carlo, estimating π by generating a large number of random points in a square. He codes this algorithm in Python and runs it with different versions of Python, logging the running time. Formally, there are a total of n released versions of Python, namely 3.1, 3.2, ..., 3.n. The running time of the algorithm on version 3.i is a_i ms. Little Q is surprised to find that the running efficiency of Python does change with version iterations.

Little Q also tests the running time of this algorithm in C++, which is stably k ms. Then he comes up with a funny idea: using the experimental data to predict which future version of Python will have a higher efficiency than C++. Unfortunately, he forgets how to apply the regression model, so he uses a brute prediction method:

- For a future version i that $i > n$, $a_i = \max(0, 2a_{i-1} - a_{i-2})$.

With this prediction method, please tell him the earliest version of Python that has a strictly higher efficiency than C++, i.e., find the minimum i such that $a_i < k$.

Input

The first line contains two integers n and k ($2 \le n \le 10$, $1 \le k \le 1000$), indicating the number of released versions of Python and the running time of the algorithm in C++.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($k < a_i \le 10^5$), indicating the running time of the algorithm on each version of Python.

Output

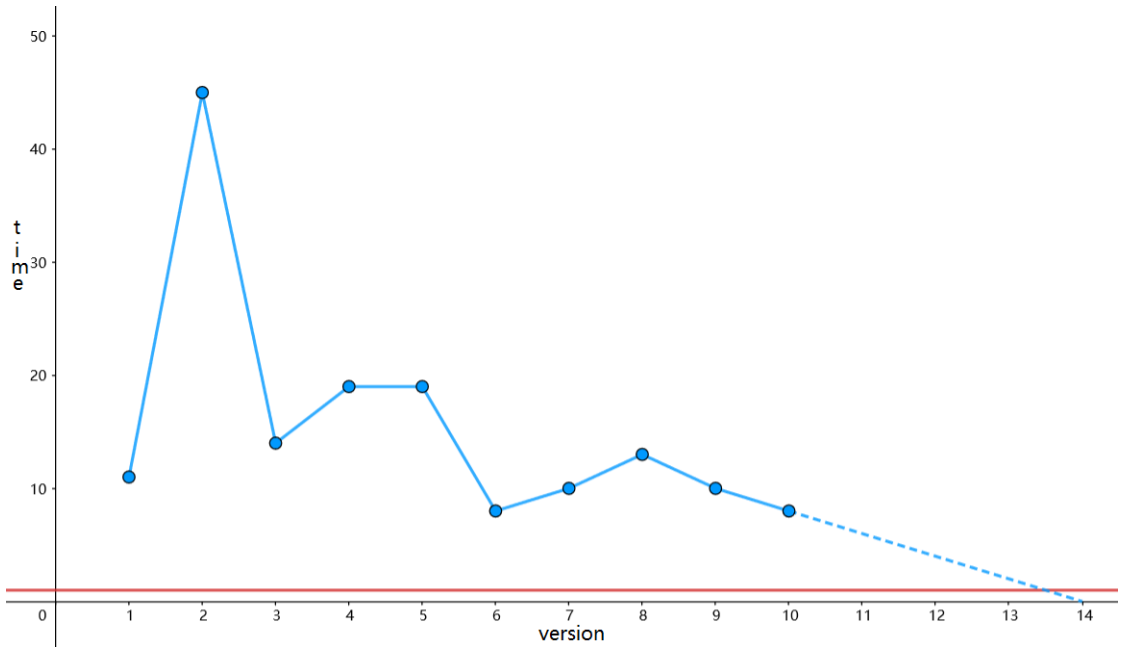
Output `Python 3.`x `will be faster than C++`, where x is an integer indicating the earliest version that $a_x < k$. If such a version does not exist, output `Python will never be faster than C++`.

Examples

standard input	standard output
10 1 11 45 14 19 19 8 10 13 10 8	Python 3.14 will be faster than C++
10 1 2 2 2 2 2 2 2 2 2 2	Python will never be faster than C++

Note

The first sample can be expressed in the following picture, in which the blue line represents the efficiency of Python, and the red line represents the efficiency of C++. The dashed line indicates the prediction. As you see, Python 3.14 will be faster than C++.



For the second sample, it is easy to notice that Python always runs for 2 ms, including the prediction of future versions. Therefore, Python will never be faster than C++, which runs for 1 ms.

Problem F. Mooncake Delivery

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

Melon is a rabbit living on the moon who is responsible for delivering moon cakes at the Mid-Autumn Festival every year. Besides the Earth, many other planets also have a demand for mooncakes. Therefore, Melon needs to consume her fairy power to travel among different planets when delivering mooncakes.

There are n planets numbered from 1 to n, along with m tunnels. Each tunnel undirectedly connects two planets. Each planet i has a color c_i and a cost w_i indicating the power that Melon needs to consume to land on it.

In order to save the power for interstellar travel, Melon will use a special skill: planting cinnamon trees. Initially, no cinnamon trees are planted on any of the planets. Once Melon lands on a planet without a cinnamon tree, she will immediately plant a cinnamon tree there, but she will not plant more than one cinnamon tree on each planet. If a planet is already planted with a cinnamon tree, Melon will not consume any power when she lands on that planet again. In addition, when she is on planet i, she can select a planet j that is planted with a cinnamon tree and is not colored c_i, and then remotely cut down the cinnamon tree on planet j and recover w_j power.

Formally, Melon can perform any one of the following actions at any time on planet u:

- Move to an adjacent planet v that is not planted with a cinnamon tree via a tunnel, consume w_v power, and plant a cinnamon tree on planet v.
- Move to an adjacent planet that is planted with a cinnamon tree via a tunnel, without consuming power and without planting additional trees.
- Select a planet v that is planted with a cinnamon tree and is not colored c_u, and then remotely cut down the cinnamon tree on planet v (i.e., no cinnamon tree is planted on planet v thereafter) and recover w_v power. Please note that this action can be performed multiple times on the same planet.

Melon's fairy power cannot be lower than 0, so she must prepare enough power before she departs. Melon wants to know the minimum amount of power she needs to prepare for the travel from the starting planet s to the terminal planet t. Specially, Melon needs to consume w_s power and plant a cinnamon tree on the starting planet s.

Input

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

The first line of each test case contains two integers n and m ($2 \le n \le 300$, $1 \le m \le \frac{n(n-1)}{2}$), indicating the number of planets and tunnels.

The second line of each test case contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le n$), indicating the color of each planet.

The third line of each test case contains n integers $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le 10^9$), indicating the power to be consumed on each planet.

Each of the following m lines contains two integers u and v ($1 \le u, v \le n$, $u \ne v$), indicating an undirected tunnel between planet u and v. It is guaranteed that there is at most one tunnel between any two different planets, and there exists a path between any two different planets.

It is guaranteed that $\sum n \le 300$ over all test cases.

Output

Output n lines, each of which contains n integers separated by spaces. The integer in the i-th row and j-th column indicates the minimum amount of power that needs to be prepared for the travel from the starting planet i to the terminal planet j. Specially, output 0 when $i = j$.

Example

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

Note

For the sample, here is a way to go from planet 3 to planet 1 with 6 initial power:

1. Consume 4 power and plant a cinnamon tree on planet 3. (Remaining power: 2)
2. Move to planet 2, consume 2 power and plant a cinnamon tree on it. (Remaining power: 0)
3. Cut down the cinnamon tree on planet 3 and recover 4 power. (Remaining power: 4)
4. Move to planet 1, consume 1 power and plant a cinnamon tree on it. (Remaining power: 3)

Problem G. Grade 2

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

Kotsuki the Cat, together with FuuFuu and Pico, lives in the KFP apartment. As a teacher, Kotsuki needs to go to school every day to give lessons. One day, Kotsuki gives a math lesson to some grade 2 students in primary school, covering the following topics:

- Coprime: Two integers are coprime if the only positive integer that is a divisor of both of them is 1.
- Bitwise XOR ($\oplus$): Bitwise XOR is a binary operation that takes two integers and performs the logical exclusive OR operation on each pair of corresponding bits of their binary forms. The result in each will be 1 if only one of the bits is 1, but will be 0 if both are 0 or both are 1. For example, $5 \oplus 3 = (101)_2 \oplus (011)_2 = (110)_2 = 6$.

After class, Kotsuki assigns homework to the students:

- Given an integer x, for different intervals $[l, r]$, please calculate the number of integers k within the interval satisfying $kx \oplus x$ and x are coprime. Formally, please calculate

$$\sum_{k=l}^{r} [\gcd(kx \oplus x, x) = 1]$$

where $\gcd(a, b)$ denotes the greatest common divisor of a and b, and $[\,]$ denotes the Iverson bracket $[P] = \begin{cases} 1 & \text{if } P \text{ is true} \\ 0 & \text{otherwise} \end{cases}$. Specially, $\gcd(0, a) = a$.

Can you solve this grade 2 homework?

Input

The first line contains two integers x ($1 \le x \le 10^6$) and n ($1 \le n \le 10^5$), where n indicates the number of intervals.

Each of the following n lines contains two integers l and r ($1 \le l \le r \le 10^{12}$), indicating an interval $[l, r]$.

Output

For each interval, output the answer in a single line.

Example

standard input	standard output
15 2 1 4 11 4514	2 2252

Note

When $x = 15$,

- $k = 1$, $\gcd(kx \oplus x, x) = \gcd(15 \oplus 15, 15) = \gcd(0, 15) = 15$
- $k = 2$, $\gcd(kx \oplus x, x) = \gcd(30 \oplus 15, 15) = \gcd(17, 15) = 1$
- $k = 3$, $\gcd(kx \oplus x, x) = \gcd(45 \oplus 15, 15) = \gcd(34, 15) = 1$
- $k = 4$, $\gcd(kx \oplus x, x) = \gcd(60 \oplus 15, 15) = \gcd(51, 15) = 3$

So the answer to interval $[1, 4]$ is 2.

Problem H. Party Animals

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

Macaronlin the Pony is a party animal who enjoys throwing parties for his friends. One day, he invites n friends to his home party. As a party expert, he prepares a lot of games for his friends, including *Multiplayer Rock-paper-scissors*.

Rock-paper-scissors is a well-known game around the world, in which each player simultaneously makes one of three gestures, including *rock*, *paper*, and *scissors*. *Rock* beats *scissors*, *scissors* beats *paper*, and *paper* beats *rock*. If both players choose the same gesture, the game is tied.

Multiplayer Rock-paper-scissors is a multiplayer version of the original game. In this game, n players are lined up in a row, numbered 1 to n from left to right. Then, the host Macaronlin will choose an interval $[l, r]$ for each round, and the players in this interval will play the game in order. Specifically, player l and $l + 1$ play first, then $l + 1$ and $l + 2$, and so on, and finally $r - 1$ and r.

Macaronlin discovers that the strategy his friends use to play the game is very simple. Initially, each player has a preferred gesture, which is one of *rock*, *paper*, and *scissors*. When two players play a game, they will both use their currently preferred gestures. As soon as a player loses a game, his preferred gesture will immediately change to the gesture that beat him. Otherwise, his preferred gesture will not change, including the case where two players choose the same gesture resulting in a tie.

Macaronlin is going to hold several rounds of the game. He wants to know what a certain player's currently preferred gesture is after some rounds. Formally, there are two types of actions:

- `1 l r`: Macaronlin chooses an interval $[l, r]$, and the players in this interval will play the game in order. Specifically, player l and $l + 1$ play first, then $l + 1$ and $l + 2$, and so on, and finally $r - 1$ and r.
- `2 x`: Macaronlin wants to know the currently preferred gesture of the player numbered x.

Please tell Macaronlin the answer for each action of the second type.

Input

The first line contains two integers n and m ($1 \le n, m \le 2 \times 10^5$), indicating the number of players and actions.

The second line contains a string of length n, consisting of `R`, `P` and `S` representing *rock*, *paper*, and *scissors* respectively. The i-th character indicates the gesture initially preferred by the player numbered i.

Each of the following m lines indicates an action, which is either `1 l r` ($1 \le l < r \le n$) or `2 x` ($1 \le x \le n$).

Output

For each action of the second type, output a character in a single line indicating the answer, which is one of `R`, `P` and `S` representing *rock*, *paper*, and *scissors* respectively.

Examples

standard input	standard output
10 10 RPSPSSRRPS 1 1 5 1 2 6 1 3 7 1 4 8 1 2 9 2 9 2 5 1 3 6 2 8 2 3	P R P R
10 10 SRPSPRPRPR 2 10 1 2 9 2 1 2 6 2 3 1 1 7 2 2 1 4 9 1 2 8 2 7	R S P S S S

Note

For the first sample, each action is shown as follows:

1. RPSPSSRRPS → PSSSSSRRPS
2. PSSSSSRRPS → PSSSSSRRPS
3. PSSSSSRRPS → PSSSSRRRPS
4. PSSSSRRRPS → PSSSRRRRPS
5. PSSSRRRRPS → PSSRRRRPPS
6. PSSRRRRP[P]S
7. PSSR[R]RRPPS
8. PSSRRRRPPS → PSRRRRRPPS
9. PSRRRRR[P]PS
10. PS[R]RRRRPPS

Problem I. Dragon Bloodline

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

Adonis the Dragon is a great leader of the dragon kingdom. It is an important mission for the dragons to pass on the dragon bloodline. Therefore, Adonis decides to choose a special day each year for egg production. Today is the day!

It is not easy to produce a dragon egg, which needs to collect various kinds of essences such as earth, water, wind, fire, etc. Specifically, the production of a dragon egg requires n kinds of essences, the amount for the i-th of which is a_i.

To improve efficiency, Adonis employs many worker dragons to collect different kinds of essences for him. There are k different levels of worker dragons, from level 1 to level k. The amount of essence that a worker dragon of level i can obtain in one day is 2^{i-1}, but each worker dragon can collect only one kind of essence. Adonis can assign which kind of essence each worker dragon should collect at the beginning, but it cannot be changed once assigned.

Knowing that the number of dragons of level i is b_i, Adonis wants to know the maximum number of dragon eggs that can be produced today with the best assignment.

Input

The first line contains an integer T ($1 \le T \le 10^3$), denoting the number of test cases.

The first line of each test case contains two integers n and k ($1 \le n \le 5 \times 10^4$, $1 \le k \le 20$), denoting the number of kinds of essences and the maximum level of worker dragons.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), denoting the amount of each kind of essence required to produce a dragon egg.

The third line of each test case contains k integers $b_1, b_2 \ldots, b_k$ ($1 \le b_i \le 10^9$), denoting the number of worker dragons of each level.

It is guaranteed that $\sum n \le 5 \times 10^4$ over all test cases.

Output

For each test case, output an integer in a single line denoting the maximum number of dragon eggs that can be produced.

Example

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

Note

Here is one possible assignment for the first case of the sample:

1. Assign 3 worker dragons of level 1 to the first essence, with a daily production of $3 \times 1 = 3$.
2. Assign 2 worker dragons of level 2 to the second essence, with a daily production of $2 \times 2 = 4$.
3. Assign 1 worker dragon of level 1, 2 worker dragons of level 2, and 1 worker dragon of level 3 to the third essence, with a daily production of $1 \times 1 + 2 \times 2 + 1 \times 4 = 9$.
4. Assign 3 worker dragons of level 3 to the fourth essence, with a daily production of $3 \times 4 = 12$.

The number of dragon eggs that can be produced is $\min(\lfloor\frac{3}{1}\rfloor, \lfloor\frac{4}{2}\rfloor, \lfloor\frac{9}{3}\rfloor, \lfloor\frac{12}{4}\rfloor) = \min(3, 2, 3, 3) = 2$.

Problem J. Eat, Sleep, Repeat

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

FuuFuu the Panda, together with Pico and Kotsuki, lives in the KFP apartment. As a rare species, FuuFuu is well-treated. Therefore, his daily activities are only eating and sleeping.

One afternoon, Kotsuki is still working outside, and FuuFuu has finished his lunch and intends to go to sleep. But Pico feels too bored to play alone every day and wants to play a game with FuuFuu for a while. Although not very reluctant, FuuFuu agrees.

Pico picks n integers $a_1, a_2, \ldots a_n$, and sets k constraints, the i-th of which is $\text{limit}_{x_i} = y_i$, indicating that the maximum number of occurrences of x_i is y_i. Then Pico and FuuFuu take turns playing the game, where each player can choose a positive integer among $a_1, a_2, \ldots a_n$ for each turn and reduce it by 1. A player will lose the game if he cannot perform any action on his turn, where there are two cases:

- No matter which of $a_1, a_2, \ldots a_n$ is chosen, there exists an integer x whose number of occurrences will be strictly greater than limit_x.
- $a_1 = a_2 = \cdots = a_n = 0$.

Even though FuuFuu is sleepy, he does not want to lose the game. Please tell him who will win if Pico goes first and both of them play optimally.

Input

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

The first line of each test case contains two integers n and k ($1 \le n \le 10^5$, $0 \le k \le 10^5$), indicating the number of integers and constraints.

The second line contains n integers $a_1, a_2, \ldots a_n$ ($0 \le a_i \le 10^9$), indicating the initial integers. It is guaranteed that the initial number of occurrences of each integer does not exceed the limit.

The i-th of the following k lines contains two integers x_i and y_i ($0 \le x_i \le 10^9$, $0 \le y_i \le n$), indicating that $\text{limit}_{x_i} = y_i$. It is guaranteed that $x_1, x_2, \ldots x_k$ are all distinct.

It's guaranteed that $\sum n \le 10^5$ and $\sum k \le 10^5$ over all test cases.

Output

For each test case, output the name of the winner in a single line, which is either `Pico` or `FuuFuu`.

Example

standard input	standard output
5 2 0 1 2 2 1 1 2 0 1 3 2 3 3 4 0 2 1 1 3 2 2 3 3 1 2 0 1 5 4 6 7 8 12 17 1 1 2 1 9 0 10 1	Pico FuuFuu Pico FuuFuu Pico

Note

For the first test case of the sample, since there are no constraints, the game ends only if all the integers are reduced to zero. So FuuFuu will lose the game after three turns.

For the second test case of the sample, the maximum number of occurrences of 0 is 1. Obviously, there must be two zeros after Pico's action in the third turn, so FuuFuu will win the game.

Problem K. I Wanna Maker

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

Redcrown the Bear is a gamer who likes playing hardcore games such as *I Wanna* series. As an experienced player, he has also made many *I Wanna* levels by himself.

Redcrown is making a new level with a puzzle: given $\tilde{k}, \tilde{x}$ and several consecutive positive integers from l to r, the player needs to determine whether there exist $\tilde{k}$ different integers such that their sum is $\tilde{x}$. Now Redcrown needs to decide the range of the given integers in this level, denoted by the interval $[l, r]$. In addition to $0 < l \leq r$, he wants to make the interval satisfy n conditions, each of which is of one of the following two types:

- `1 k x`: there exist k different integers within the interval $[l, r]$ such that their sum is x.
- `2 k x`: there do not exist k different integers within the interval $[l, r]$ such that their sum is x. (There are two possible cases: there are less than k integers in this interval; or, there are k or more integers in this interval, but there are no k integers whose sum is x.)

Redcrown wants to know the number of different intervals $[l, r]$ that satisfy the conditions.

Input

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

The first line of each test case contains an integer n ($1 \leq n \leq 10^5$), indicating the number of conditions.

Each of the following n lines contains three integers t, k, x ($1 \leq t \leq 2$, $1 \leq k \leq 10^9$, $1 \leq x \leq 10^9$), indicating a condition.

It is guaranteed that $\sum n \leq 10^5$ over all test cases.

Output

For each test case, output an integer in a single line indicating the number of different intervals that satisfy the conditions. If there are an infinite number of intervals, output -1 in a single line.

Example

standard input	standard output
4 2 1 1 2 2 1 4 2 1 1 4 2 1 2 2 1 1 1 2 1 1 4 2 1 15 1 5 20 1 3 8 2 2 25	4 -1 0 7

Note

For the first test case of the sample, there are two conditions:

1. There exists an integer 2 within the interval.
2. There does not exist an integer 4 within the interval.

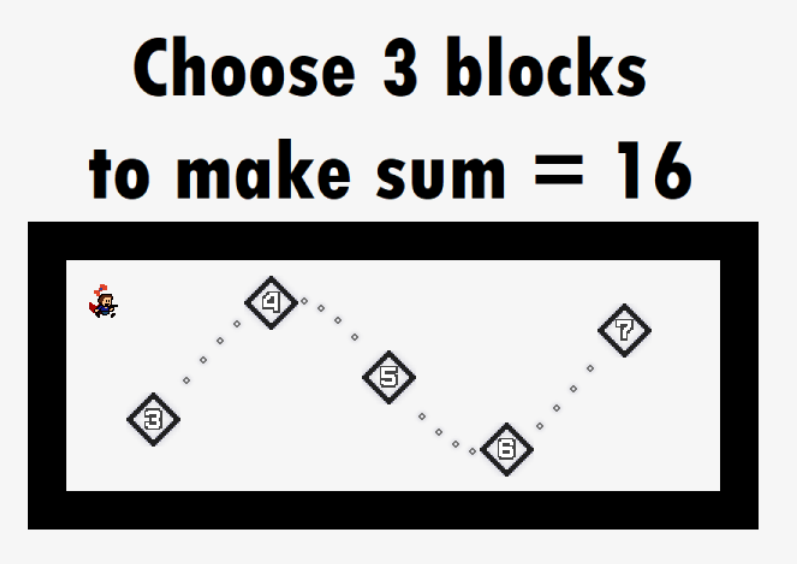
There are four intervals that satisfy the conditions: $[1, 2]$, $[1, 3]$, $[2, 2]$ and $[2, 3]$.

For the second test case of the sample, there are two conditions:

1. There exists an integer 4 within the interval.
2. There does not exist an integer 2 within the interval.

All intervals with $3 \leq l \leq 4$ and $r \geq 4$ satisfy the conditions, so there are an infinite number of intervals.

Here is a reference picture of the *I Wanna* level:



Problem L. Novice Magician

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

HoshiYo the Fox is a magician who is new to the magic school. As a novice, he is not proficient in many kinds of magic, especially those about numbers.

One day, HoshiYo learns a kind of magic that can change the integers in an array of length 2^n. However, due to a lack of proficiency, he can only change half of the integers in the array each time he uses magic. Formally, let's denote the array as $a_0, a_1, \ldots, a_{2^n-1}$, the following action can be performed by using the magic once:

- Choose 2^{n-1} different integers in the array and an integer x (possibly negative), add x to one of them, add $x+2$ to another one of them, add $x+4$ to another one of them, ..., and add $x+2^n-2$ to the last remaining one. In other words, choose 2^{n-1} different indices ranging from 0 to 2^n-1 denoted by $p_0, p_1, \ldots, p_{2^{n-1}-1}$, and add $x+2i$ to a_{p_i} for each integer i within $0 \le i < 2^{n-1}$.

Initially, each integer in the array is 0. HoshiYo has an ideal array $b_0, b_1, \ldots, b_{2^n-1}$ in his mind. He wonders if he can get this ideal array by using magic at most 2^n times, i.e., to make $a_0 = b_0, a_1 = b_1, \ldots, a_{2^n-1} = b_{2^n-1}$.

Input

The first line contains an integer n ($1 \le n \le 11$), indicating that the length of the array is 2^n.

The second line contains 2^n integers $b_0, b_1, \ldots, b_{2^n-1}$ ($0 \le b_i \le 10^5$), indicating the ideal array.

Output

If the ideal array cannot be obtained by using magic at most 2^n times, output `NO` in a single line.

Otherwise, output `YES` in the first line and an integer k ($0 \le k \le 2^n$) in the second line indicating the number of times to use magic. Then output k lines, each of which contains $2^{n-1}+1$ integers $x, p_0, p_1, \ldots, p_{2^{n-1}-1}$ separated by spaces. You need to ensure that $p_0, p_1, \ldots, p_{2^{n-1}-1}$ are distinct integers between 0 and 2^n-1, and each of $a_0, a_1, \ldots, a_{2^n-1}$ is always in the range of $[-10^{18}, 10^{18}]$.

If there are multiple solutions, output any.

Examples

standard input	standard output
2 1 14 5 14	YES 4 0 1 0 2 1 2 12 1 3 -1 0 2
2 11 45 1 4	NO

Note

For the first sample, the result after each use of magic is as follows:

1. $\{0+2, 0+0, 0, 0\} = \{2, 0, 0, 0\}$
2. $\{2, 0+2, 0+4, 0\} = \{2, 2, 4, 0\}$
3. $\{2, 2+12, 4, 0+14\} = \{2, 14, 4, 14\}$
4. $\{2-1, 14, 4+1, 14\} = \{1, 14, 5, 14\}$

Problem M. String Master

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

Rousong the Fox is a string master who can handle a wide variety of strings with ease. For example, he can braid noodles with his tongue. Likewise, he is good with string data types in computer programming.

One night, Rousong has a dream, in which he picks an infinitely long string out of the bowl when eating noodles. On closer inspection, it is a binary string containing only zeros and ones, concatenated by $0, 1, 10, 11, \ldots$. Formally, let's define string $s^0 = 0$, and $s^i = s^{i-1} + (i)_2$ for each integer $i > 0$, where $a + b$ denotes the concatenation of string a and b, and $(i)_2$ denotes the binary form of integer i without leading zeros. Consequently, the infinitely long string Rousong dreamed of is $s^\infty = 011011100101\ldots$.

Since the length of s^∞ is too large, Rousong only wants to focus on the substring from the l-th character to the r-th character, denoted as $s^\infty_{l,r}$. He wants to find the substring of length n within the string $s^\infty_{l,r}$ with the largest lexicographical order. Formally, please find the index i within $l \le i \le r - n + 1$ that maximizes the lexicographical order of $s^\infty_{i,i+n-1}$. Can you help Rousong solve this problem before he wakes up?

For two binary strings of the same length a and b, we say that string a is lexicographically greater than string b if for the first index i where a and b differ, the i-th character of a is 1 and of b is 0.

Input

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

Each test case contains three integers l, r, n ($1 \le l \le r \le 10^{18}$, $1 \le n \le \min(r - l + 1, 10^6)$) in a single line, indicating the range to be focused on and the length of the substring.

It is guaranteed that $\sum n \le 10^6$ over all test cases.

Output

For each test case, output the substring with the largest lexicographical order in a single line.

Example

standard input	standard output
6 6 13 3 1 9 9 1 1451419198 10 987 6543 21 1123 581321 34 1000010 1000030 18	110 011011100 1111111111 11111111100000000010 111111111111111100000000000000100 000111110011010000

Note

For the first test case of the sample, $s^\infty = 01101\underline{1100101}110111\ldots$, so $s^\infty_{6,13} = 11001011$. The substrings of length 3 are $110, 100, 001, 010, 101, 011$. The one with the largest lexicographical order is 110.