A. Shashliks

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

You are the owner of a popular shashlik restaurant, and your grill is the heart of your kitchen. However, the grill has a peculiarity: after cooking each shashlik, its temperature drops.

You need to cook as many portions of shashlik as possible, and you have an unlimited number of portions of two types available for cooking:

- The first type requires a temperature of at least a degrees at the start of cooking, and after cooking, the grill's temperature decreases by x degrees.
- The second type requires a temperature of at least b degrees at the start of cooking, and after cooking, the grill's temperature decreases by y degrees.

Initially, the grill's temperature is k degrees. Determine the maximum total number of portions of shashlik that can be cooked.

Note that the grill's temperature can be negative.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The only line of each test case contains five integers k, a, b, x, and y ($1 \le k, a, b, x, y \le 10^9$) — the initial temperature of the grill, the required temperature for cooking the first and second types of shashlik, respectively, as well as the temperature drop after cooking the first and second types of shashlik, respectively.

Output

For each test case, output a single integer — the maximum number of portions of shashlik that you can cook.

Standard Input	Standard Output
5 10 3 4 2 1 1 10 10 1 1 100 17 5 2 3 28 14 5 2 4 277 5 14 1 3	8 0 46 10 273

Note

In the first test case, it is advantageous to cook 7 portions of the second type of shashlik; after that, the grill's temperature will be 3 degrees, and we can cook one more portion of the first type of shashlik.

In the second test case, we cannot cook either type of shashlik because the grill is too cold.

In the fourth test case, it is advantageous to cook 8 portions of the first type of shashlik; after that, it will be possible to cook another 2 portions of the second type of shashlik.

,

B. Good Start

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

The roof is a rectangle of size $w \times h$ with the bottom left corner at the point $(0, 0)$ on the plane. Your team needs to completely cover this roof with identical roofing sheets of size $a \times b$, with the following conditions:

- The sheets cannot be rotated (not even by 90°).
- The sheets must not overlap (but they can touch at the edges).
- The sheets can extend beyond the boundaries of the rectangular roof.

A novice from your team has already placed two such sheets on the roof in such a way that the sheets **do not overlap** and each of them **partially covers the roof**.

Your task is to determine whether it is possible to completely tile the roof without removing either of the two already placed sheets.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains four integers w, h, a, and b ($1 \le w, h, a, b \le 10^9$) — the dimensions of the roof and the dimensions of the roofing sheets, respectively.

The second line of each test case contains four integers x_1, y_1, x_2, and y_2 ($-a + 1 \le x_1, x_2 \le w - 1, -b + 1 \le y_1, y_2 \le h - 1$) — the coordinates of the bottom left corners of the already placed roofing sheets. It is guaranteed that these sheets do not overlap.

Output

For each test case, output "Yes" (without quotes) if it is possible to completely tile the roof without removing either of the two already placed tiles, and "No" (without quotes) otherwise.

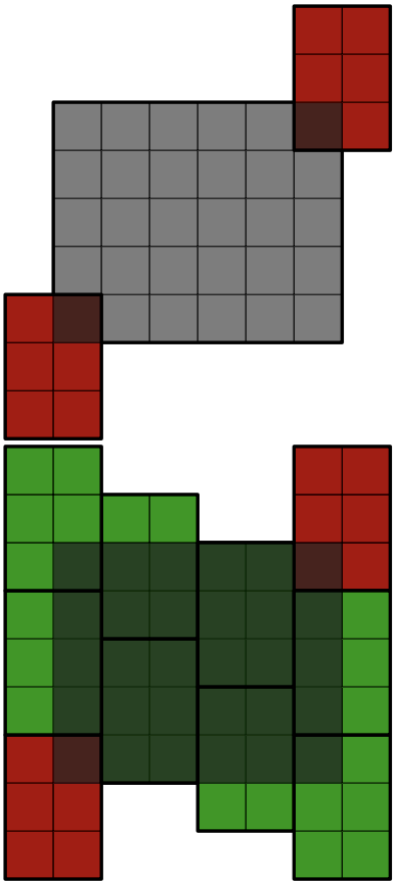
You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

Standard Input	Standard Output
7 6 5 2 3 -1 -2 5 4 4 4 2 2 0 0 3 1 10 9 3 2 0 0 4 3 10 9 3 2 0 0 6 3 5 5 2 2 -1 -1 4 -1 5 5 2 2	Yes No No Yes No Yes No

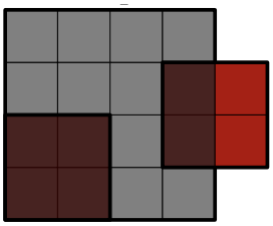
-1 -1 2 3 7 8 2 4 0 0 0 5	

Note

In the first test case, it is possible to add 8 roofing sheets as follows:



In the second test case, it is impossible to completely tile the roof:



,

C. Smilo and Minecraft

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

The boy Smilo is playing Minecraft! To prepare for the battle with the dragon, he needs a lot of golden apples, and for that, he requires a lot of gold. Therefore, Smilo goes to the mine.

The mine is a rectangular grid of size $n \times m$, where each cell can be either gold ore, stone, or an empty cell. Smilo can blow up dynamite in any empty cell. When dynamite explodes in an empty cell with coordinates (x, y), all cells within a square of side $2k + 1$ centered at cell (x, y) become empty. If gold ore was located **strictly inside** this square (not on the boundary), it disappears. However, if the gold ore was on the boundary of this square, Smilo collects that gold.

Dynamite can only be detonated inside the mine, but the explosion square can extend beyond the mine's boundaries.

Determine the maximum amount of gold that Smilo can collect.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains three integers n, m, and k ($1 \le n, m, k \le 500$) — the number of rows, columns, and the explosion parameter k, respectively.

Each of the following n lines contains m characters, each of which is equal to `'.'`, `'#'`, or `'g'`, where `'.'` — is an empty cell, `'#'` — is stone, `'g'` — is gold. It is guaranteed that at least one of the cells is empty.

It is guaranteed that the sum $n \cdot m$ across all test cases does not exceed $2.5 \cdot 10^5$.

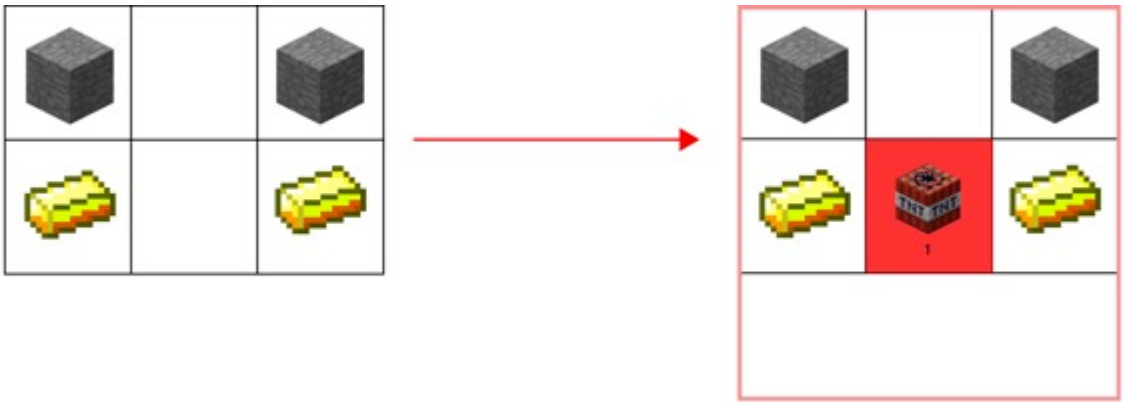
Output

For each test case, output a single integer — the maximum amount of gold that can be obtained.

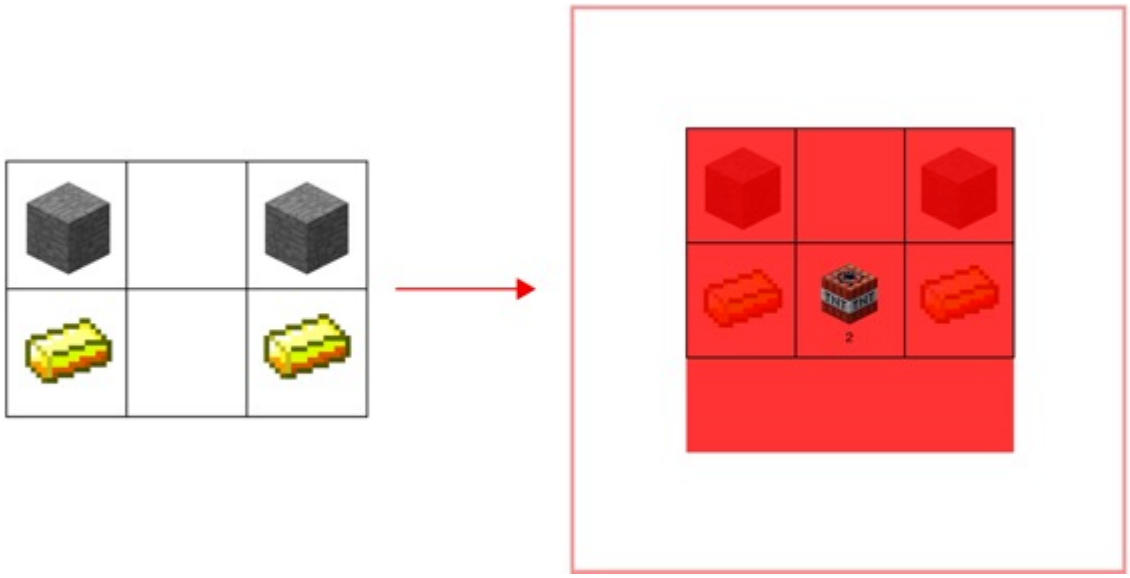
Standard Input	Standard Output
3 2 3 1 #.# g.g 2 3 2 #.# g.g 3 4 2 .gg. g..# g##.	2 0 4

Note

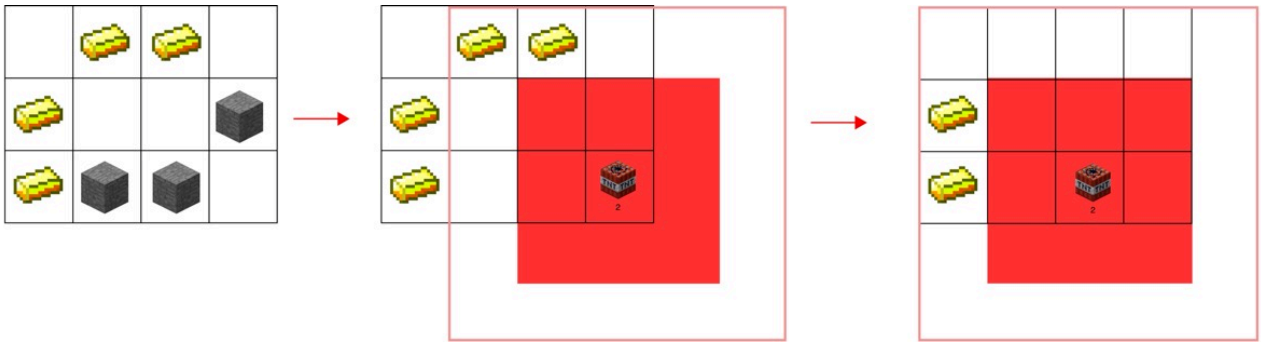
In the first test case, Smilo can detonate the dynamite in any empty cell and obtain 2 gold:



In the second test case, no matter what Smilo does, he will not be able to obtain any gold:



In the third test case, it is possible to detonate the dynamite in the bottom right corner to obtain 2 gold, and then make another explosion one cell to the left to obtain the remaining 2 gold:



,

D. Cheater

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

You are playing a new card game in a casino with the following rules:

1. The game uses a deck of $2n$ cards with different values.
2. The deck is evenly split between the player and the dealer: each receives n cards.
3. Over n rounds, the player and the dealer simultaneously play one top card from their hand. The cards are compared, and the point goes to the one whose card has a higher value. The winning card is removed from the game, while the losing card is returned to the hand **and placed on top of the other cards** in the hand of the player who played it.

Note that the game always lasts **exactly** n rounds.

You have tracked the shuffling of the cards and know the order of the cards in the dealer's hand (from top to bottom). You want to maximize your score, so you can swap any two cards in your hand **no more than once** (to avoid raising suspicion).

Determine the maximum number of points you can achieve.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 5 \cdot 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the number of cards in the player's hand.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2n$) — the values of the cards in the player's hand from top to bottom.

The third line of each test case contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le 2n$) — the values of the cards in the dealer's hand from top to bottom.

It is guaranteed that the values of all cards are distinct.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer — the maximum number of points you can achieve.

Standard Input	Standard Output
3 7 13 7 4 9 12 10 2 6 1 14 3 8 5 11 3 1 6 5 2 3 4	6 2 3

5 8 6 3 10 1 7 9 5 2 4	

Note

In the first test case, the cards can remain unchanged. The gameplay will be organized as follows:

1. The cards with values 13 and 6 are compared. The player wins and receives a point.
2. The cards with values 7 and 6 are compared. The player wins and receives a point.
3. The cards with values 4 and 6 are compared. The dealer wins.
4. The cards with values 4 and 1 are compared. The player wins and receives a point.
5. The cards with values 9 and 1 are compared. The player wins and receives a point.
6. The cards with values 12 and 1 are compared. The player wins and receives a point.
7. The cards with values 10 and 1 are compared. The player wins and receives a point.

Thus, the player received 6 points.

In the second test case, the cards with values 1 and 5 can be swapped, then the player's hand looks as follows $[5, 6, 1]$. The gameplay will be organized as follows:

1. The cards with values 5 and 2 are compared. The player wins and receives a point.
2. The cards with values 6 and 2 are compared. The player wins and receives a point.
3. The cards with values 1 and 2 are compared. The dealer wins.

Thus, the player received 2 points.

In the third test case, the cards with values 3 and 10 can be swapped, then the player's hand looks as follows $[8, 6, 10, 3, 1]$. The gameplay will be organized as follows:

1. The cards with values 8 and 7 are compared. The player wins and receives a point.
2. The cards with values 6 and 7 are compared. The dealer wins.
3. The cards with values 6 and 9 are compared. The dealer wins.
4. The cards with values 6 and 5 are compared. The player wins and receives a point.
5. The cards with values 10 and 5 are compared. The player wins and receives a point.

Thus, the player received 3 points.

,

E. From Kazan with Love

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

Marat is a native of Kazan. Kazan can be represented as an undirected tree consisting of n vertices. In his youth, Marat often got into street fights, and now he has m enemies, numbered from 1 to m, living in Kazan along with him.

Every day, all the people living in the city go to work. Marat knows that the i-th of his enemies lives at vertex a_i and works at vertex b_i. He himself lives at vertex x and works at vertex y. It is guaranteed that $a_i \neq x$.

All enemies go to work via the shortest path and leave their homes at time 1. That is, if we represent the shortest path between vertices a_i and b_i as $c_1, c_2, c_3, \ldots, c_k$ (where $c_1 = a_i$ and $c_k = b_i$), then at the moment p ($1 \leq p \leq k$), the enemy numbered i will be at vertex c_p.

Marat really does not want to meet any of his enemies at the same vertex at the same time, as this would create an awkward situation, but they **can meet on an edge**. Marat also leaves his home at time 1, and at each subsequent moment in time, he can either move to an adjacent vertex or stay at his current one.

Note that Marat can only meet the i-th enemy at the moments $2, 3, \ldots, k$ (where $c_1, c_2, \ldots, c_k$ is the shortest path between vertices a_i and b_i). In other words, starting from the moment after the enemy reaches work, Marat **can no longer meet him**.

Help Marat find the earliest moment in time when he can reach work without encountering any enemies along the way, or determine that it is impossible.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains four integers n, m, x, and y ($2 \leq n \leq 10^5$, $1 \leq m \leq 200$, $1 \leq x, y \leq n$, $x \neq y$) — the number of vertices in the tree, the number of enemies, and the vertex numbers from which Marat starts his journey and where he needs to arrive, respectively.

The j-th of the following $n - 1$ lines contains two integers v_j and u_j ($1 \leq v_j, u_j \leq n$, $v_j \neq u_j$) — the endpoints of the j-th edge of the tree.

The i-th of the following m lines contains two integers a_i and b_i ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$, $a_i \neq x$) — the description of the routes of Marat's enemies.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output a single integer — the minimum moment in time when Marat can reach work, or -1 if it is impossible.

Standard Input	Standard Output
5 4 1 1 4	4 6

1 2 2 3 3 4 4 1 5 1 1 5 1 2 2 3 3 4 4 5 5 1 9 2 1 9 1 2 2 3 3 4 3 5 5 6 6 7 6 8 8 9 9 1 7 1 9 2 7 2 1 4 2 5 3 6 4 5 5 6 4 7 5 8 6 9 2 8 3 7 3 2 1 3 1 2 2 3 2 1 3 1	10 5 -1

Note

In the first test case, it is possible to reach vertex number 4 from vertex number 1 via the shortest path. Note that Marat will meet a single enemy on an edge, not at a vertex.

In the second test case, the optimal strategy is to wait for one moment in time at the starting vertex and then go along the shortest path from vertex 1 to vertex 5. If he does not stop at the beginning, Marat will meet his enemy at a vertex, not on an edge.

In the third test case, it is beneficial to go from vertex 1 to vertex 4. After that, he should not move anywhere for one moment in time, and then go along the shortest path from vertex 4 to vertex 9.

,

F. Two Arrays

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

You are given two arrays a and b of length n. You can perform the following operation an unlimited number of times:

- Choose an integer i from 1 to n and swap a_i and b_i.

Let $f(c)$ be the number of distinct numbers in array c. Find the maximum value of $f(a) + f(b)$. Also, output the arrays a and b after performing all operations.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$) — the length of the arrays.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2n$) — the elements of array a.

The third line of each test case contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le 2n$) — the elements of array b.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, print a single integer in the first line — the maximum value of $f(a) + f(b)$.

In the second line, print n integers — the elements of array a after performing the operations.

In the third line, print n integers — the elements of array b after performing the operations.

Standard Input	Standard Output
3 5 1 2 4 4 4 1 3 3 5 2 7 2 2 4 4 5 5 5 1 3 3 2 1 6 6 7 12 3 3 4 5 6 4 1 2 13 8 10 13 7	9 1 3 4 5 2 1 2 3 4 4 12 2 3 4 2 1 5 6 1 2 3 4 5 6 5 14 12 3 13 8 10 6 4 1 2 3 4 5 13 7

Note

In the first test case, after applying three operations with $i = 2$, $i = 4$, and $i = 5$, we obtain $a = [1, 3, 4, 5, 2]$ and $b = [1, 2, 3, 4, 4]$. After that, $f(a) + f(b) = 5 + 4 = 9$. It can be shown that it is not possible to achieve a greater answer.

In the second test case, after applying the operations:

$$f([2,3,4,2,1,5,6]) + f([1,2,3,4,5,6,5]) = 6 + 6 = 12$$