A. Concatenation of Arrays

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

You are given n arrays $a_1, \ldots, a_n$. The length of each array is two. Thus, $a_i = [a_{i,1}, a_{i,2}]$. You need to concatenate the arrays into a single array of length $2n$ such that the number of inversions[†] in the resulting array is minimized. Note that you **do not need** to count the actual number of inversions.

More formally, you need to choose a permutation[‡] p of length n, so that the array $b = [a_{p_1,1}, a_{p_1,2}, a_{p_2,1}, a_{p_2,2}, \ldots, a_{p_n,1}, a_{p_n,2}]$ contains as few inversions as possible.

[†] The number of inversions in an array c is the number of pairs of indices i and j such that $i < j$ and $c_i > c_j$.

[‡] A permutation of length n is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2, 3, 1, 5, 4]$ is a permutation, but $[1, 2, 2]$ is not a permutation (2 appears twice in the array), and $[1, 3, 4]$ is also not a permutation ($n = 3$ but there is 4 in the array).

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. 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 number of arrays.

Each of the following n lines contains two integers $a_{i,1}$ and $a_{i,2}$ ($1 \le a_{i,j} \le 10^9$) — the elements of the i-th array.

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

Output

For each test case, output $2n$ integers — the elements of the array you obtained. If there are multiple solutions, output any of them.

Standard Input	Standard Output

4 2 1 4 2 3 3 3 2 4 3 2 1 5 5 10 2 3 9 6 4 1 8 7 1 10 20	2 3 1 4 2 1 3 2 4 3 4 1 2 3 5 10 8 7 9 6 10 20

Note

In the first test case, we concatenated the arrays in the order $2, 1$. Let's consider the inversions in the resulting array $b = [2, 3, 1, 4]$:

- $i = 1$, $j = 3$, since $b_1 = 2 > 1 = b_3$;
- $i = 2$, $j = 3$, since $b_2 = 3 > 1 = b_3$.

Thus, the number of inversions is 2. It can be proven that this is the minimum possible number of inversions.

In the second test case, we concatenated the arrays in the order $3, 1, 2$. Let's consider the inversions in the resulting array $b = [2, 1, 3, 2, 4, 3]$:

- $i = 1$, $j = 2$, since $b_1 = 2 > 1 = b_2$;
- $i = 3$, $j = 4$, since $b_3 = 3 > 2 = b_4$;
- $i = 5$, $j = 6$, since $b_5 = 4 > 3 = b_6$.

Thus, the number of inversions is 3. It can be proven that this is the minimum possible number of inversions.

In the third test case, we concatenated the arrays in the order $4, 2, 1, 5, 3$.

,

B. Skipping

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

It is already the year 3024, ideas for problems have long run out, and the olympiad now takes place in a modified individual format. The olympiad consists of n problems, numbered from 1 to n. The i-th problem has its own score a_i and a certain parameter b_i ($1 \le b_i \le n$).

Initially, the testing system gives the participant the **first** problem. When the participant is given the i-th problem, they have two options:

- They can submit the problem and receive a_i points;
- They can skip the problem, in which case they will never be able to submit it.

Then, the testing system selects the next problem for the participant from problems with indices j, such that:

- If he submitted the i-th problem, it looks at problems with indices $j < i$;
- If he skipped the i-th problem, it looks at problems with indices $j \le b_i$.

Among these problems, it selects the problem with the **maximum** index that it has **not previously given** to the participant (he has neither submitted nor skipped it before). If there is no such problem, then the competition for the participant **ends**, and their result is equal to the sum of points for all submitted problems. In particular, if the participant submits the first problem, then the competition for them ends. Note that the participant receives each problem **at most once**.

Prokhor has prepared thoroughly for the olympiad, and now he can submit any problem. Help him determine the maximum number of points he can achieve.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 10^5$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 4 \cdot 10^5$) — the number of problems in the olympiad.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the scores of the problems.

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

It is guaranteed that the sum of n over all test cases does not exceed $4 \cdot 10^5$.

Output

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

Standard Input	Standard Output
4 2 15 16	16 200

2 1 5 10 10 100 100 1000 3 4 1 1 1 3 100 49 50 3 2 2 4 100 200 300 1000 2 3 4 1	100 1000

Note

In the first test case, Prokhor can skip the first problem; then he will receive the problem with index $b_1 = 2$. Prokhor can submit it and receive $a_2 = 16$ points. After that, the competition will end because Prokhor has already received all problems. Note that if Prokhor submits the first problem, he will receive $a_1 = 15$ points, but the competition will end immediately.

In the second test case, Prokhor can skip the first problem; then he will receive the problem with index $b_1 = 3$. Prokhor can submit it and receive $a_3 = 100$ points. After that, Prokhor will receive the second problem, which he can skip to receive the problem with index $b_2 = 4$. Prokhor can submit the fourth problem and receive another $a_4 = 100$ points. After that, the competition ends because Prokhor has already received all problems with indices not exceeding 4. Thus, Prokhor will receive a total of 200 points.

In the third test case, Prokhor can submit the first problem and receive 100 points, after which the competition will end immediately.

,

C. C+K+S

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

You are given two strongly connected† directed graphs, each with exactly n vertices, but possibly different numbers of edges. Upon closer inspection, you noticed an important feature — the length of any cycle in these graphs is divisible by k.

Each of the $2n$ vertices belongs to exactly one of two types: *incoming* or *outgoing*. For each vertex, its type is known to you.

You need to determine whether it is possible to draw exactly n directed edges between the source graphs such that the following four conditions are met:

- The ends of any added edge lie in different graphs.
- From each outgoing vertex, exactly one added edge originates.
- Into each incoming vertex, exactly one added edge enters.
- In the resulting graph, the length of any cycle is divisible by k.

† A strongly connected graph is a graph in which there is a path from every vertex to every other vertex.

Input

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

The first line of each test case contains two integers n and k ($2 \le k \le n \le 2 \cdot 10^5$) — the number of vertices in each graph and the value by which the length of each cycle is divisible.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($a_i \in \{0, 1\}$). If $a_i = 0$, then vertex i of the first graph is incoming. If $a_i = 1$, then vertex i of the first graph is outgoing.

The third line of each test case contains a single integer m_1 ($1 \le m_1 \le 5 \cdot 10^5$) — the number of edges in the first graph.

The next m_1 lines contain descriptions of the edges of the first graph. The i-th of them contains two integers v_i and u_i ($1 \le v_i, u_i \le n$) — an edge in the first graph leading from vertex v_i to vertex u_i.

Next, in the same format, follows the description of the second graph.

The next line contains n integers $b_1, b_2, \ldots, b_n$ ($b_i \in \{0, 1\}$). If $b_i = 0$, then vertex i of the second graph is incoming. If $b_i = 1$, then vertex i of the second graph is outgoing.

The next line contains a single integer m_2 ($1 \le m_2 \le 5 \cdot 10^5$) — the number of edges in the second graph.

The next m_2 lines contain descriptions of the edges of the second graph. The i-th of them contains two integers v_i and u_i ($1 \le v_i, u_i \le n$) — an edge in the second graph leading from vertex v_i to vertex u_i.

It is guaranteed that both graphs are strongly connected, and the lengths of all cycles are divisible by k.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$. It is guaranteed that the sum of m_1 and the sum of m_2 over all test cases does not exceed $5 \cdot 10^5$.

Output

For each test case, output "YES" (without quotes) if it is possible to draw n new edges such that all conditions are met, and "NO" (without quotes) otherwise.

You may output the answer in any case (for example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as a positive answer).

Standard Input	Standard Output
3 4 2 1 0 0 1 4 1 2 2 3 3 4 4 1 1 0 0 1 4 1 3 3 2 2 4 4 1 3 3 0 0 0 3 1 2 2 3 3 1 1 1 0 3 1 2 2 3 3 1 4 2 1 1 1 1 4 1 2 2 3 3 4 4 1 0 0 0 0 6 1 2 2 1 1 3 3 1 1 4 4 1	YES NO YES

Note

In the first test case, it is possible to draw edges from the first graph to the second graph as $(1, 3)$ and $(4, 2)$ (the first number in the pair is the vertex number in the first graph, and the second number in the pair is the vertex number in the second graph), and from the second graph to the first graph as $(1, 2)$, $(4, 3)$ (the first number in the pair is the vertex number in the second graph, and the second number in the pair is the vertex number in the first graph).

In the second test case, there are a total of 4 incoming vertices and 2 outgoing vertices, so it is not possible to draw 3 edges.

,

D. Many Games

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

Recently, you received a rare ticket to the only casino in the world where you can actually earn something, and you want to take full advantage of this opportunity.

The conditions in this casino are as follows:

- There are a total of n games in the casino.
- You can play each game **at most once**.
- Each game is characterized by two parameters: p_i ($1 \le p_i \le 100$) and w_i — the probability of winning the game in percentage and the winnings for a win.
- If you lose in any game you decide to play, you will receive nothing at all (even for the games you won).

You need to choose a set of games in advance that you will play in such a way as to maximize the expected value of your winnings.

In this case, if you choose to play the games with indices $i_1 < i_2 < \ldots < i_k$, you will win in all of them with a probability of $\prod\limits_{j=1}^{k} \frac{p_{i_j}}{100}$, and in that case, your winnings will be equal to $\sum\limits_{j=1}^{k} w_{i_j}$.

That is, the expected value of your winnings will be $\left(\prod\limits_{j=1}^{k} \frac{p_{i_j}}{100}\right) \cdot \left(\sum\limits_{j=1}^{k} w_{i_j}\right)$.

To avoid going bankrupt, the casino owners have limited the expected value of winnings for each individual game. Thus, for all i ($1 \le i \le n$), it holds that $w_i \cdot p_i \le 2 \cdot 10^5$.

Your task is to find the maximum expected value of winnings that can be obtained by choosing some set of games in the casino.

Input

The first line contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the number of games offered to play.

The i-th of the following n lines contains two integers p_i and w_i ($1 \le p_i \le 100$, $1 \le w_i, p_i \cdot w_i \le 2 \cdot 10^5$) — the probability of winning and the size of the winnings in the i-th game.

Output

Output a single number — the maximum expected value of winnings in the casino that can be obtained by choosing some subset of games.

Your answer will be accepted if the relative or absolute error does not exceed 10^{-6}. Formally, if a is your answer and b is the jury's answer, it will be accepted if $\frac{|a-b|}{\max(b,1)} \le 10^{-6}$.

Standard Input	Standard Output
3 80 80 70 100	112.00000000

50 200	
2 100 1 100 1	2.00000000
4 1 100 2 1000 2 100 3 1	20.00000000
5 34 804 78 209 99 191 61 439 90 79	395.20423800

Note

In the first example, you can choose the first and third games. In this case, the expected value of winnings will be $\left(\frac{p_1}{100} \cdot \frac{p_3}{100}\right) \cdot (w_1 + w_3) = \left(\frac{80}{100} \cdot \frac{50}{100}\right) \cdot (80 + 200) = 112$.

In the second example, you can choose the first and second games. In this case, the expected value of winnings will be $\left(\frac{p_1}{100} \cdot \frac{p_2}{100}\right) \cdot (w_1 + w_2) = \left(\frac{100}{100} \cdot \frac{100}{100}\right) \cdot (1 + 1) = 2$.

In the third example, you can choose only the second game. In this case, the expected value of winnings will be $\frac{p_2}{100} \cdot w_2 = \frac{2}{100} \cdot 1000 = 20$.

,

E. Tree of Life

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

In the heart of an ancient kingdom grows the legendary Tree of Life — the only one of its kind and the source of magical power for the entire world. The tree consists of n nodes. Each node of this tree is a magical source, connected to other such sources through magical channels (edges). In total, there are $n - 1$ channels in the tree, with the i-th channel connecting nodes v_i and u_i. Moreover, there exists a unique simple path through the channels between any two nodes in the tree.

However, the magical energy flowing through these channels must be balanced; otherwise, the power of the Tree of Life may disrupt the natural order and cause catastrophic consequences. The sages of the kingdom discovered that when two magical channels converge at a single node, a dangerous "magical resonance vibration" occurs between them. To protect the Tree of Life and maintain its balance, it is necessary to select several paths and perform special rituals along them. A path is a sequence of distinct nodes $v_1, v_2, \ldots, v_k$, where each pair of adjacent nodes v_i and v_{i+1} is connected by a channel. When the sages perform a ritual along such a path, the resonance vibration between the channels (v_i, v_{i+1}) and (v_{i+1}, v_{i+2}) is blocked for each $1 \le i \le k - 2$.

The sages' task is to select the minimum number of paths and perform rituals along them to block all resonance vibrations. This means that for every pair of channels emanating from a single node, there must exist **at least one** selected path that contains **both** of these channels.

Help the sages find the minimum number of such paths so that the magical balance of the Tree of Life is preserved, and its power continues to nourish the entire world!

Input

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

The first line of each test case contains a single integer n ($2 \le n \le 5 \cdot 10^5$) — the number of nodes in the Tree of Life.

The i-th of the following $n - 1$ lines of each test case contains two integers v_i and u_i ($1 \le v_i < u_i \le n$) — the channel connecting nodes v_i and u_i.

It is guaranteed that there exists a unique simple path through the channels between any two nodes.

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

Output

For each test case, output a single integer — the minimum number of paths that the sages need to select to prevent a catastrophe.

Standard Input	Standard Output
5 4 1 2 2 3	1 0 3

3 4 2 1 2 4 1 2 1 3 1 4 8 3 7 2 4 1 2 2 5 3 6 1 3 3 8 6 2 3 1 2 3 6 1 5 1 4	7 3

Note

In the first test case, there are two pairs of channels emanating from a single node: $(1, 2)$ and $(2, 3)$, $(2, 3)$ and $(3, 4)$. It is sufficient to perform the ritual along the path $1 - 2 - 3 - 4$. Thus, the answer is 1.

In the second test case, there are no pairs of channels emanating from a single node, so the answer is 0.

In the third test case, rituals can be performed along the paths $2 - 1 - 3$, $2 - 1 - 4$, and $3 - 1 - 4$.

,

F. Hills and Pits

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 512 megabytes

In a desert city with a hilly landscape, the city hall decided to level the road surface by purchasing a dump truck. The road is divided into n sections, numbered from 1 to n from left to right. The height of the surface in the i-th section is equal to a_i. If the height of the i-th section is greater than 0, then the dump truck must take sand from the i-th section of the road, and if the height of the i-th section is less than 0, the dump truck must fill the pit in the i-th section of the road with sand. It is guaranteed that the initial heights are not equal to 0.

When the dump truck is in the i-th section of the road, it can either take away x units of sand, in which case the height of the surface in the i-th section will decrease by x, or it can fill in x units of sand (provided that it currently has at least x units of sand in its bed), in which case the height of the surface in the i-th section of the road will increase by x.

The dump truck can start its journey from any section of the road. Moving to an adjacent section on the left or right takes 1 minute, and the time for loading and unloading sand can be neglected. The dump truck has an infinite capacity and is initially empty.

You need to find the minimum time required for the dump truck to level the sand so that the height in each section becomes equal to 0. Note that after all movements, the dump truck **may still have sand left in its bed**. You need to solve this problem **independently** for the segments numbered from l_i to r_i. Sand outside the segment cannot be used.

Input

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

The first line of each test case contains two integers n and q ($1 \le n, q \le 3 \cdot 10^5$) — the number of sections and the number of queries.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$, $a_i \neq 0$) — the initial height in each section.

The i-th of the following q lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le n$) — the boundaries of the segment of sections for which the minimum time needs to be determined.

It is guaranteed that the sum of n over all test cases and the sum of q over all test cases do not exceed $3 \cdot 10^5$.

Output

For each query, output the minimum time required to level the sand in the segment $[l_i, r_i]$, or -1 if it is impossible.

Standard Input	Standard Output
5 1 1 -179 1 1	-1 2 5 -1

5 3 -2 2 -1 3 -1 2 4 1 5 1 3 7 1 1 1 1 -4 1 1 1 1 7 7 2 2 -2 2 -2 1 2 -1 1 7 2 7 4 4 1000000000 1000000000 999999999 -1000000000 2 4 3 4 2 3 1 3	8 6 6 2 -1 1 2

Note

In the first test case, 179 units of sand need to be added to the only section. However, there is nowhere to take it from, so this is impossible.

In the second test case:

- In the first query, the dump truck can start its journey at the second section. It can take 2 units of sand, after which the height in the second section will become 0. Then the dump truck can move to the third section. It can pour 1 unit of sand there, after which the height in the third section will become 0. Then the dump truck can move to the fourth section. There it can take 3 units of sand, after which the height in the fourth section will become 0. In total, the dump truck will spend 2 minutes on movements.
- In the second query, the dump truck can start its journey at the fourth section. It can take 3 units of sand, after which the height in the fourth section will become 0. Then the dump truck can move to the fifth section. It can pour 1 unit of sand there, after which the height in the fifth section will become 0. Then the dump truck can move back to the fourth section and then to the third. It can pour 1 unit of sand there, after which the height in the third section will become 0. Then the dump truck can move to the second section. It can take 2 units of sand. Then it can move to the first section. It can pour 2 units of sand there, after which the height in the first section will become 0. In total, the dump truck will spend 5 minutes on movements.
- In the third query, the dump truck will not be able to make the height in each section equal to 0.