The 2nd Universal Cup

Stage 11: Nanjing

November 25-26, 2023

This problem set should contain 13 problems on 28 numbered pages.

Based on

International Collegiate Programming Contest (ICPC)

Hosted by

Prepared by

Problem A. Cool, It's Yesterday Four Times More

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

After the great success in 2018, 2019, 2020, 2021 and 2022, Nanjing University of Aeronautics and Astronautics (NUAA) will host the *International Collegiate Programming Contest* (ICPC) Nanjing regional for the sixth time in a row.

Team ***Power of Two*** and team ***Three Hold Two*** won the champion title for Tsinghua University in 2018 and 2019. In 2020, 2021 and 2022, team ***Inverted Cross*** from Peking University won the three-peat champion titles. This year, there are around 330 teams participating in the contest. There are at most 33 gold medals, 66 silver medals and 99 bronze medals that will be awarded (note that these numbers are for reference only). We are looking forward to seeing participants' outstanding performance!

What's even better is that, as the pandemic has come to an end, we can finally gather in Nanjing to participate in this wonderful contest. We'd like to be grateful for the hard work done by all staff and volunteers for this contest. Thank you all for your great contribution to this contest!

The 2018 ICPC Asia Nanjing Regional Contest

In the 2018 contest, problem K, ***Kangaroo Puzzle***, requires the contestants to construct an operation sequence for the game:

> The puzzle is a grid with n rows and m columns ($1 \le n, m \le 20$) and there are some (at least 2) kangaroos standing in the puzzle. The player's goal is to control them to get together. There are some walls in some cells and the kangaroos cannot enter the cells with walls. The other cells are empty. The kangaroos can move from an empty cell to an adjacent empty cell in four directions: up, down, left, and right.
>
> There is exactly one kangaroo in every empty cell in the beginning and the player can control the kangaroos by pressing the button U, D, L, R on the keyboard. The kangaroos will move simultaneously according to the button you press.
>
> The contestant needs to construct an operating sequence of at most 5×10^4 steps consisting of U, D, L, R only to achieve the goal.

In the 2020 contest, problem A, ***Ah, It's Yesterday Once More***, requires the contestants to construct an input map to hack the following code of the problem described before:

```
#include <bits/stdc++.h>
using namespace std;
string s = "UDLR";
int main()
{
  srand(time(NULL));
  for (int i = 1; i <= 50000; i++) putchar(s[rand() % 4]);
  return 0;
}
```

In the 2021 contest, problem A, ***Oops, It's Yesterday Twice More***, also requires the contestants to construct an operation sequence for the game:

> This time, every cell in the grid stands exactly one kangaroo. You need to construct an operating sequence consisting only of characters 'U', 'D', 'L', and 'R'. After applying it, you must make sure every kangaroo will gather at the specific cell (a, b). The length of the operating sequence cannot exceed $3(n-1)$. As always, the kangaroos will move simultaneously according to the operation you command.

In the 2022 contest, problem A, ***Stop, Yesterday Please No More***, asks the contestants to solve the following counting problem:

> This time, every cell (except one which is a hole) in the grid stands exactly one kangaroo. The operating sequence is given and all kangaroos stepping out of the grid or onto the hole will be removed. Given the number of kangaroos remaining after all operations, count the number of positions which might be the hole.

Now, in the 2023 contest, the kangaroo problem is back again! We don't know why problem setters are so obsessed with kangaroos but the problem is as follows:

You are given a grid with n rows and m columns. Each cell is either a hole or empty. In each empty cell stands exactly one kangaroo.

Similarly, the kangaroos are controlled by pressing the button U, D, L, R on the keyboard. All kangaroos will move simultaneously according to the button pressed. Specifically, for any kangaroo located in the cell on the i-th row and the j-th column, indicated by (i, j):

1. Button U: it will move to $(i-1, j)$.
2. Button D: it will move to $(i+1, j)$.
3. Button L: it will move to $(i, j-1)$.
4. Button R: it will move to $(i, j+1)$.

If a kangaroo steps onto a hole or steps out of the grid, it will be removed from the grid. If after applying a sequence of operations (possibly an empty sequence) there is exactly one kangaroo remaining on the grid, that kangaroo becomes the winner.

The problem is: for each kangaroo, determine if there exists a sequence of operations to make it the winner. Output the total number of kangaroos which are possible to become the winner.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n, m \le 10^3$, $1 \le n \times m \le 10^3$) indicating the number of rows and columns of the grid.

For the following n lines, the i-th line contains a string $s_{i,1}s_{i,2}\cdots s_{i,m}$ of length m where each character is either '.' (dot, ascii: 46) or 'O' (capitalized letter, ascii: 79). If $s_{i,j}$ is a '.' then grid (i, j) is empty; If $s_{i,j}$ is a 'O' then grid (i, j) is a hole.

It's guaranteed that the sum of $n \times m$ of all test cases will not exceed 5×10^3.

Output

For each test case output one line containing one integer indicating the number of kangaroos for which there exists a sequence of operations to make it the winner.

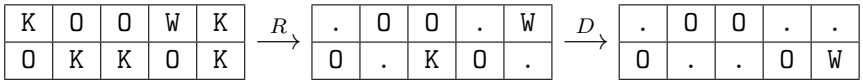
Example

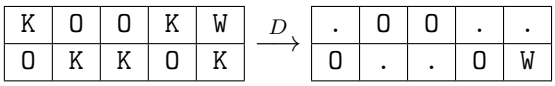
standard input	standard output
4 2 5 .OO.. O..O. 1 3 O.O 1 3 .O. 2 3 OOO OOO	3 1 0 0

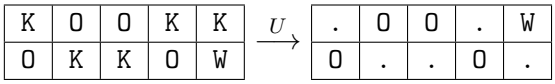
Note

The sample test cases are explained below. We use 'W' to indicate the kangaroo which later becomes winner and 'K' to indicate the other kangaroos.

For the first sample test case, kangaroos initially located at $(1, 4)$, $(1, 5)$ and $(2, 5)$ may become winners. Possible sequences of operations are shown as follows:







For the second sample test case, as there is only one kangaroo, no operation is needed for it to become the winner.

For the third sample test case, as any operation will remove the two kangaroos at the same time, there is no possible winner.

For the fourth sample test case, as there is no kangaroo, there is no possible winner.

Problem B. Intersection over Union

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	1024 megabytes

Intersection over Union, also known as the *Jaccard index* and the *Jaccard similarity* coefficient (originally given the French name coefficient de communaut é by Paul Jaccard), is a statistic used for gauging the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Intersection over Union (IOU) is also a widely used metric in computer vision for evaluating object detection and segmentation algorithms.

A rectangle that is aligned with the x and y axes (i.e., its sides are parallel to the x and y axes) is often called an "axis-aligned rectangle", "axis-aligned bounding box" (AABB), or simply "bounding box". On the other hand, a rectangle that is not necessarily aligned with the x and y axes (i.e., its sides might be at an angle) is often called a "rotated rectangle", a "rotated bounding box", or an "oriented bounding box" (OBB). In computer vision and image processing applications, both types of rectangles are widely in use, depending on the problem at hand.

In this problem, your task is to find an axis-aligned rectangle (AABB) that maximizes the IOU with a rotated rectangle (OBB). The IOU between two rectangles is defined as the area of the intersection between the two rectangles divided by the area of their union.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^4$) indicating the number of test cases. For each test case:

The first and only line contains eight integers x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4 ($-10^9 \le x_i, y_i \le 10^9$), where (x_i, y_i) represents the coordinates of the i-th vertex of the rotated rectangle, in either clockwise or counterclockwise order. It's guaranteed that the rotated rectangle has a positive area.

Output

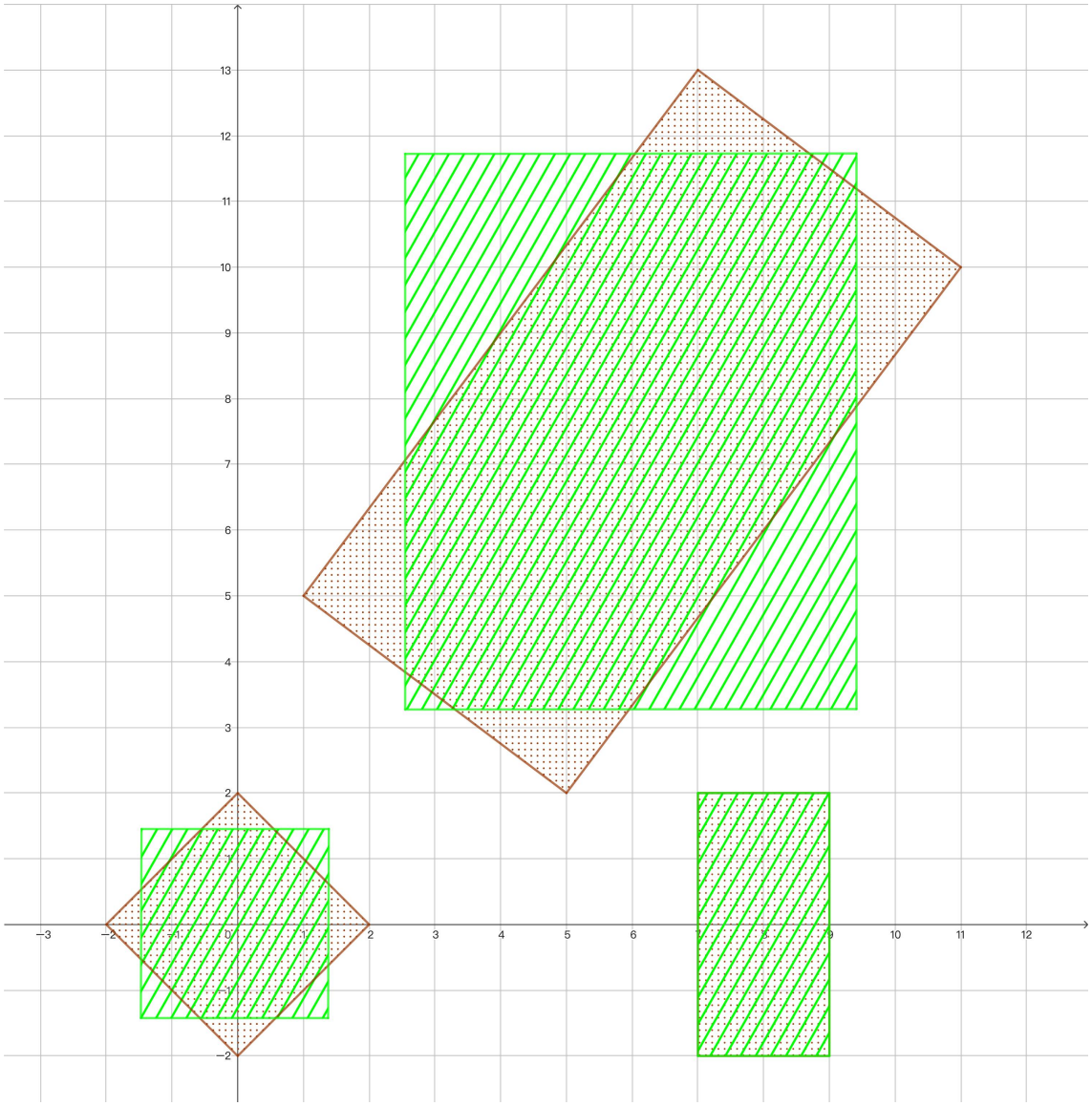
For each test case, output one line containing one number indicating the maximum IOU between the rotated rectangle and the axis-aligned rectangle. Your answer will be considered correct if the absolute or relative error does not exceed 10^{-9}.

Example

standard input	standard output
3 0 2 2 0 0 -2 -2 0 7 -2 9 -2 9 2 7 2 7 13 11 10 5 2 1 5	0.70710678118654752 1 0.62384322483109367

Note

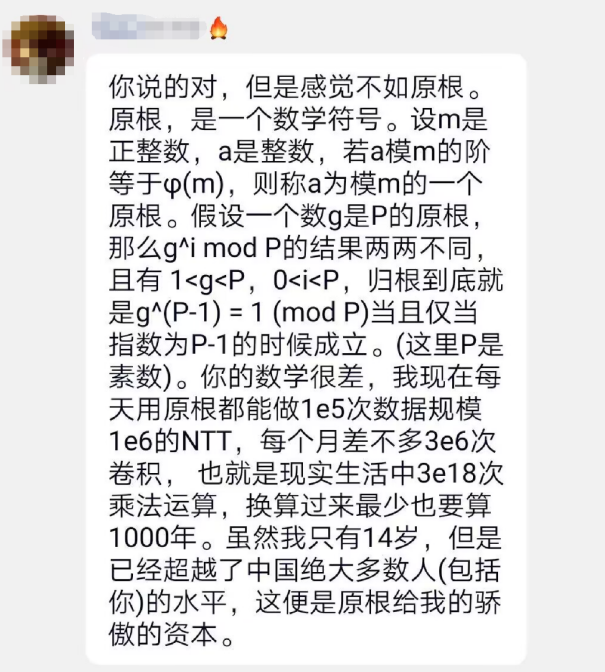
The sample test cases are shown as follows. The input rectangles are shown with dots and the optimal axis-aligned rectangles are shown with hatching.



Problem C. Primitive Root

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

BaoBao has just learnt about primitive roots in number theory and is showing off his knowledge to Little Cyan Fish through an instant messaging software.



This image is only for amusement and has nothing to do with the problem itself. You can safely skip this image if you can't read Chinese.

Based on the fact that if a non-negative integer g is a primitive root modulo P (where P is a prime), then $g^{P-1} \equiv 1 \pmod{P}$, BaoBao decided to use the expression `(g ^ (P - 1)) % P == 1` to check if g is a primitive root modulo P. Unfortunately, in most programming languages (for example C and C++), `^` is the bitwise exclusive-or (XOR) operator, not the power operator. Little Cyan Fish spotted this issue at once and now he is interested in the following problem:

Given a prime number P and a non-negative integer m, how many non-negative integers g satisfies $g \le m$ and $g \oplus (P-1) \equiv 1 \pmod{P}$? Here $\oplus$ is bitwise exclusive-or (XOR) operator.

Please help Little Cyan Fish solve this problem.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^5$) indicating the number of test cases. For each test case:

The first and only line contains two integers P and m ($2 \le P \le 10^{18}$, $0 \le m \le 10^{18}$, P is a prime).

Output

For each test case, output one line containing one integer indicating the number of g satisfying the constraints.

Example

standard input	standard output
3 2 0 7 11 1145141 998244353	1 2 872

Problem D. Red Black Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

There is a rooted tree with n vertices numbered from 1 to n, where vertex 1 is the root. Each vertex has a color, which is either red or black.

We say a vertex is good, if every simple path from that vertex to any of its descendant leaf vertex contains the same number of black vertices. We say a tree is perfect, if all its vertices are good.

Let R_k be the subtree rooted at vertex k. For each $1 \le k \le n$, answer the following query: If you can choose a set of vertices and change their colors (that is, change red vertices to black, and change black vertices to red), calculate the minimum number of vertices you have to choose to make R_k perfect.

Recall that a simple path is a path which does not go through the same edge multiple times.

Also recall that a subtree rooted at vertex k is a tree consisting of all descendants of vertex k and has vertex k as the root. Note that any vertex is a descendant of itself.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \le n \le 10^5$) indicating the number of vertices in the tree.

The second line contains a string $s_1 s_2 \cdots s_n$ of length n ($s_i \in \{0, 1\}$). If $s_i = 0$ then vertex i is red; If $s_i = 1$ then vertex i is black.

The third line contains $(n-1)$ integers $p_2, p_3, \cdots, p_n$ ($1 \le p_i < i$) where p_i is the parent of vertex i.

It's guaranteed that the sum of n of all test cases will not exceed 10^6.

Output

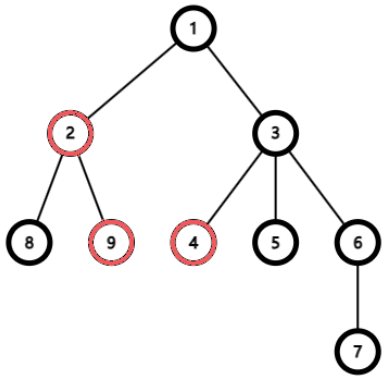
For each test case output one line containing n integers separated by a space, where the i-th integer indicates the minimum number of vertices whose color you have to change to make R_i perfect.

Please, DO NOT output extra spaces at the end of each line, or your solution may be considered incorrect!

Example

standard input	standard output
2 9 101011110 1 1 3 3 3 6 2 2 4 1011 1 1 3	4 1 2 0 0 0 0 0 0 2 0 0 0

Note



We illustrate the first sample test case.

To make R_1 perfect, we can change the color of vertices 2, 4, 6 and 9. After that, all paths from vertex 1 to its descendant leaves will contain 3 black vertices, all paths from vertex 2 to its descendant leaves will contain 2 black vertices, and all paths from vertex 3 to its descendant leaves will contain 2 black vertices. As vertices 4 to 9 only has one descendant leaf, they're always good.

To make R_2 perfect, we can change the color of vertex 8. After that, all paths from vertex 2 to its descendant leaves will contain 0 vertices. As vertices 8 and 9 only has one descendant leaf, they're always good.

Problem E. Extending Distance

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Given $n \times m$ points arranged into n rows and m columns, there are weighted bidirectional edges between adjacent points. That is, let (i, j) be the point on the i-th row and the j-th column, for all $1 \le i, i' \le n$ and $1 \le j, j' \le m$, there is an edge between (i, j) and (i', j') if and only if $|i - i'| + |j - j'| = 1$.

BaoBao will begin his journey at any point $(p, 1)$ on the first column and finish the journey at any point (q, m) on the last column. He can travel along the edges in both directions. The distance of a path is defined as the sum of the weight of the edges in the path. Among all the paths from the first column to the last column, BaoBao will choose the shortest path.

Little Cyan Fish hopes BaoBao to better enjoy the journey, so he tries to increase the weight of some edges before BaoBao begins traveling. Specifically, Little Cyan Fish can increase the weight of any edge by 1 in one operation. He hopes the distance of BaoBao's path can increase by k after all operations. Please help him calculate the minimum number of operations needed and output the corresponding solution.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains three integers n, m and k ($1 \le n \times m \le 500$, $2 \le n, m \le 500$, $1 \le k \le 100$) indicating the number of rows and columns, and the increment on the length of the shortest path.

For the following n lines, the i-th line contains $(m-1)$ integers $r_{i,1}, r_{i,2}, \cdots, r_{i,m-1}$ ($1 \le r_{i,j} \le 10^9$) where $r_{i,j}$ indicates the weight of the edge between (i, j) and $(i, j+1)$.

For the following $(n-1)$ lines, the i-th line contains m integers $c_{i,1}, c_{i,2}, \cdots, c_{i,m}$ ($1 \le c_{i,j} \le 10^9$) where $c_{i,j}$ indicates the weight of the edge between (i, j) and $(i+1, j)$.

It is guaranteed that the sum of $n \times m$ of all test cases will not exceed 5×10^3.

Output

For each test case:

First output one line containing one integer indicating the minimum number of operations needed.

Then output n lines. The i-th line contains $(m-1)$ integers $r'_{i,1}, r'_{i,2}, \cdots, r'_{i,m-1}$ ($1 \le r'_{i,j} \le 2 \times 10^9$) separated by a space, where $r'_{i,j}$ indicates the weight of the edge between (i, j) and $(i, j+1)$ after all operations.

Then output $(n-1)$ lines. The i-th line contains m integers $c'_{i,1}, c'_{i,2}, \cdots, c'_{i,m}$ ($1 \le c'_{i,j} \le 2 \times 10^9$) separated by a space, where $c'_{i,j}$ indicates the weight of the edge between (i, j) and $(i+1, j)$ after all operations.

If there are multiple valid answers, output any of them.

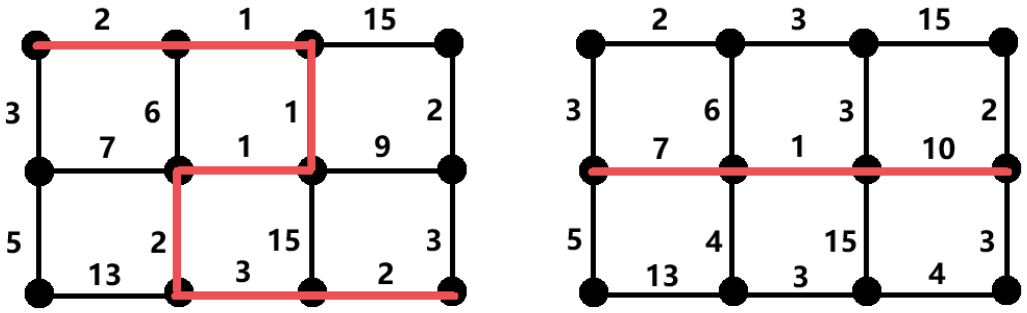
Please, DO NOT output extra spaces at the end of each line, or your solution may be considered incorrect!

Example

standard input	standard output
2 3 4 6 2 1 15 7 1 9 13 3 2 3 6 1 2 5 2 15 3 3 3 3 1 1 2 2 3 3 1 1 1 2 2 2	9 2 3 15 7 1 10 13 3 4 3 6 3 2 5 4 15 3 4 4 1 3 2 3 3 1 1 1 2 2 2

Note

The first sample test case is illustrated below. The graph on the left is the original graph and the graph on the right is the graph after increasing the weight of some edges. Shortest paths of each graph are marked by red lines.



Problem F. Equivalent Rewriting

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

There is a sequence A of length m where all elements equal to 0. We will then execute n operations on A in order. The i-th operation can be denoted as p_i distinct integers $b_{i,1}, b_{i,2}, \cdots, b_{i,p_i}$, indicating that we'll change the value of the $b_{i,j}$-th element in the sequence to i for all $1 \le j \le p_i$. Let R be the resulting sequence after all operations.

We now require you to rearrange the operations but still produce the same result. More formally, let $q_1, q_2, \cdots, q_n$ be a permutation of n that differs from $1, 2, \cdots, n$. You'll execute the q_1-th, q_2-th, ..., q_n-th operation on sequence A in order, and the final resulting sequence must equal to R. Your task is to find such permutation or state that it does not exist.

Recall that a permutation of n is a sequence of length n in which each integer from 1 to n (both inclusive) appears exactly once. Let $x_1, x_2, \cdots, x_n$ and $y_1, y_2, \cdots, y_n$ be two permutations of n. We say they're different if there exists an integer k such that $1 \le k \le n$ and $x_k \ne y_k$.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n, m \le 10^5$) indicating the number of operations and the length of the sequence.

For the following n lines, the i-th line first contains an integer p_i ($1 \le p_i \le m$) indicating the number of elements changed by the i-th operation. Then p_i distinct integers $b_{i,1}, b_{i,2}, \cdots, b_{i,p_i}$ follow ($1 \le b_{i,j} \le m$) indicating the index of the elements to be changed.

It is guaranteed that the sum of $(n + m)$ of all test cases will not exceed 2×10^6, and the sum of p_i of all test cases will not exceed 10^6.

Output

For each test case, if such permutation exists, first output `Yes` in one line. Then output n integers $q_1, q_2, \cdots, q_n$ separated by a space in the second line indicating the answer. If there are multiple valid answers, you can output any of them.

If there is no such permutation, simply output `No` in one line.

Please, DO NOT output extra spaces at the end of each line, or your solution may be considered incorrect!

Example

standard input	standard output
3 3 6 3 3 1 5 2 5 3 2 2 6 2 3 3 1 3 2 2 3 1 1 3 2 2 1	Yes 3 1 2 No No

Note

For the first sample test case, by executing the operations in either order of $\{1, 2, 3\}$ or $\{3, 1, 2\}$ yields the same resulting sequence $\{1, 3, 2, 0, 2, 3\}$.

Problem G. Knapsack

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Little Cyan Fish, an inexperienced businessman, recently launched a store named *Queen's Organic Jewelry*. This jewelry store houses n gemstones, where the i-th gemstone is priced at w_i dollar and has a beauty of v_i. Prior to visiting the store, you have W dollars in hand, which you plan to use to purchase gemstones of the greatest possible total beauty.

Interestingly, Little Cyan Fish's store is running a promotion today. Any visitor to the store can select any k gemstones and take them home absolutely free of charge! With this opportunity at hand, you're keen to know the maximum total beauty of gemstones you could obtain with your W dollars, assuming you adopt the optimal strategy.

Please bear in mind that the store stocks only one unit of each gemstone, so you cannot obtain the same gemstone more than once. Also note that you don't have to spend all the money.

Input

There is only one test case in each test file.

The first line of the input contains three integers n, W and k ($1 \le n \le 5 \times 10^3$, $1 \le W \le 10^4$, $0 \le k \le n$), indicating the total number of gemstones in the store, the amount of money you have and the number of gemstones you can take for free.

For the following n lines, the i-th line contains two integers w_i and v_i ($1 \le w_i \le W$, $1 \le v_i \le 10^9$), indicating the price and the beauty of the i-th gemstone.

Output

Output one line containing one integer indicating the answer.

Examples

standard input	standard output
4 10 1 9 10 10 1 3 5 5 20	35
5 13 2 5 16 5 28 7 44 8 15 8 41	129

Note

In the first example, Little Cyan Fish's shop holds 4 gemstones and you are permitted to take 1 gemstone for free. One optimal strategy involves taking the first gemstone for free, and purchasing the third and fourth gemstones.

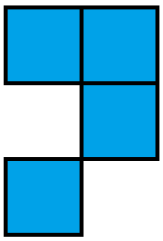
Gemstone	**Price** w_i	**Beauty** v_i	**Action**
1	9	10	Take for free
2	10	1	/
3	3	5	Purchase
4	5	20	Purchase

So the answer is $10 + 5 + 20 = 35$.

Problem H. Puzzle: Question Mark

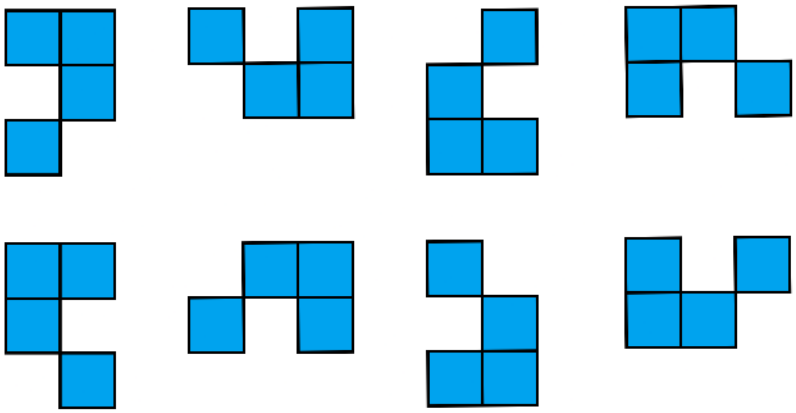
Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

XingHui is a puzzle master. Today, she is playing a puzzle called *Filling with Question Marks*. The puzzle contains a grid of n rows and n columns, and a handful of *question mark* pieces (QM pieces). A QM piece occupies 4 cells, as shown in the following figure.



A QM piece (It looks like a question mark, I suppose?)

QM pieces must be placed entirely inside the grid and pieces can be rotated by a multiple of 90 degrees and/or flipped. More precisely, there are 8 types of QM pieces, as shown below.



Any two QM pieces cannot occupy the same cell. The goal of the puzzle is to find a way to place the maximum number of QM pieces on the $n \times n$ grid.

XingHui wonders if you could solve the puzzle successfully.

Input

There are multiple test cases. The first line of the input contains a single integer T indicating the number of test cases. For each test case:

The first and only line contains one integer n ($1 \le n \le 2 \times 10^3$) indicating the size of the grid.

It is guaranteed that the sum of n^2 of all test cases will not exceed 5×10^6.

Output

For each test case:

First output one line containing one integer indicating the maximum number of QM pieces placed on the grid.

Then output n lines. Each of these lines contains n integers separated by a space. The j-th integer of the i-th line $a_{i,j}$ indicates that the cell on the i-th row and j-th column belongs to the $a_{i,j}$-th QM piece. If $a_{i,j}$ is 0, it indicates that the corresponding cell is empty and does not belong to any QM piece.

If there are multiple solutions, output any of them.

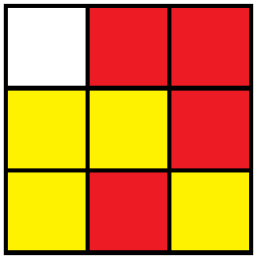
Please, DO NOT output extra spaces at the end of each line, or your solution may be considered incorrect!

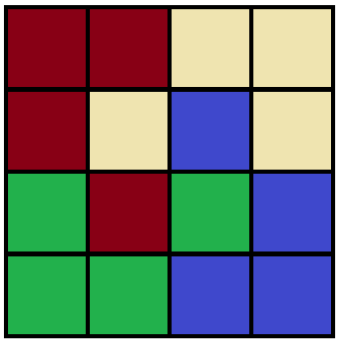
Example

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

Note

The sample test cases are shown below.





Problem I. Counter

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

There is a counter with two buttons. Pressing the "+" button will increase the value on the counter by 1 and pressing the "c" button will set the value on the counter to 0. The initial value on the counter is 0.

Someone has performed n operations on the counter. Each operation is to press one of the two buttons. There are m known conditions where the i-th condition can be described as two integers a_i and b_i, indicating that after the a_i-th operation the value on the counter is b_i.

Is there a way to press the buttons so that all known conditions are satisfied?

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n \le 10^9$, $1 \le m \le 10^5$) indicating the number of operations and the number of known conditions.

For the following m lines, the i-th line contains two integers a_i and b_i ($1 \le a_i \le n$, $0 \le b_i \le 10^9$) indicating that after the a_i-th operation the value on the counter is b_i.

It's guaranteed that the sum of m of all test cases will not exceed 5×10^5.

Output

For each test case output one line. If there exists a way to press the buttons so that all known conditions are satisfied, output `Yes`. Otherwise output `No`.

Example

standard input	standard output
3 7 4 4 0 2 2 7 1 5 1 3 2 2 2 3 1 3 1 3 100	Yes No No

Note

For the first sample test case, pressing buttons in the order of "++cc+c+" can satisfy all known conditions.

For the second sample test case, there are 8 ways to press the buttons 3 times.

Presses	2-nd Op. Result	3-rd Op. Result		Presses	2-nd Op. Result	3-rd Op. Result
ccc	0	0		+cc	0	0
cc+	0	1		+c+	0	1
c+c	1	0		++c	2	0
c++	1	2		+++	2	3

There is no way to satisfy all known conditions.

For the third sample test case, pressing the buttons 3 times can only make the value on the counter at most 3. It can't be 100.

Problem J. Suffix Structure

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

For a string $u = u_1 \dots u_n$, let $\text{pre}(u, i)$ be the prefix $u_1 \dots u_i$. In particular, $\text{pre}(u, 0)$ is empty string.

For two strings $u = u_1 \dots u_n$ and $v = v_1 \dots v_m$, let $u + v$ be the concatenation $u_1 \dots u_n v_1 \dots v_m$.

You are given a string $t = t_1 \dots t_m$ of length m and a tree T with $(n + 1)$ vertices labeled with $0, 1, \dots, n$ rooted at vertex 0. Each edge is associated with a character. Please note that in this problem, the alphabet may contain more than 26 characters.

Consider the following function

$$f(i, j) = \max\{d(x) \mid s_x \text{ is a suffix of } s_i + \text{pre}(t, j)\}$$

where s_i be the concatenation of characters on the shortest path from root to vertex i and $d(i)$ be the number of edges on the shortest path from the root to vertex i.

Your task is to compute the values of $g_1, g_2, \dots, g_m$ where $g_j = \sum_{i=1}^{n} f(i, j)$.

Note that s_0 is the empty string and empty string is a suffix of any string.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n, m \le 2 \times 10^5$).

The second line contains n integers $p_1, p_2, \dots, p_n$ ($0 \le p_i < i$) where p_i indicates the parent of vertex i.

The third line contains n integers $c_1, c_2, \dots, c_n$ ($1 \le c_i \le n$) where c_i indicates that the edge from vertex p_i to vertex i is associated with the c_i-th character from the alphabet. It is guaranteed that $p_i \neq p_j$ or $c_i \neq c_j$ for all $i \neq j$.

The fourth line contains m integers $t_1, t_2, \dots, t_m$ ($1 \le t_i \le n$) where t_i is the i-th character of string t.

It is guaranteed that neither the sum of n nor the sum of m will exceed 2×10^5.

Output

For each test case output one line containing m integers $g_1, g_2, \dots, g_m$ separated by a space.

Please, DO NOT output extra spaces at the end of each line, or your solution may be considered incorrect!

Example

standard input	standard output
2 11 3 0 1 2 0 4 5 4 6 0 9 10 1 3 2 2 1 3 4 1 3 2 1 3 2 4 5 16 0 0 0 1 4 1 2 3 2 2 2 1 3 3 2 1 3 2 1 3 2 2 1 1 2 1	17 26 22 8 5 5 5 5 5 5 5 5 5 5 5 5 5 10 5

Note

Let's calculate $f(11,1)$ and $f(11,2)$ in the first sample test case to help you further understand. We have $s_{11} = \{3,2,1\}$ so $s_{11} + \text{pre}(t,1) = \{3,2,1,3\}$. As $s_6 = \{2,1,3\}$ is its longest suffix existing in the tree, $f(11,1) = d(6) = 3$. Also $s_{11} + \text{pre}(t,2) = \{3,2,1,3,2\}$ and $s_3 = \{1,3,2\}$ is its longest suffix existing in the tree, so $f(11,2) = d(3) = 3$.

Problem K. Grand Finale

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

In Pigeland, *Slay the Pig* is the most popular roguelike game where players use card decks to challenge a boss called Evil Potato Lord.

The general game rules are as follows:

1. Players start with a set of initial cards in their hand and a draw pile arranged from top to bottom.
2. At any given time, the player's hand and the draw pile are the only places where cards are present.
3. Players can play cards from their hands, and each played card will lead to this card being discarded first, and then triggering certain effects.
4. The player can only use the next card after all the effects of the previously played card have been triggered.

In this problem, for simplicity, we will only consider the effect of drawing cards. The rules for drawing cards are as follows:

1. When using cards that can draw some other cards, players will draw a certain number of cards from the top of the draw pile into their hands in sequential order.
2. Players have a hand size limit of k, and at any given moment, the number of cards in a player's hand cannot exceed the limitation k.
3. When a player attempts to draw a card from the top of the draw pile and their hand already contains k cards, the drawn card will be discarded from the draw pile without triggering any effects and not added to the hand.
4. When a player attempts to draw a card but the draw pile is empty, nothing will happen.

Decks based on the key card *Grand Finale* are the most powerful deck in the game. Once this card is played, it will cause massive damage to all enemies and often leads to victory in the game. However, to play the Grand Finale card, there are strict conditions. That is, the player's draw pile must be empty, which means that all cards must either be played, discarded, or in the player's hand.

Bie-Bot is the smartest pig in Pigeland only after the Mysterious Oscar, and he is also playing a *Grand Finale*-based deck, which can contain the following four types of cards:

1. *Grand Finale*: The most powerful card in the game, playing it will lead to an easy victory, and there is exactly one *Grand Finale* in Bie-Bot's deck. This card can only be played if there are no cards in the draw pile.

2. *Quick Slash*: Using this card, you will draw one card from the draw pile.
3. *Backflip*: Using this card, you will draw two cards from the draw pile.
4. *Wound*: A status card that, once it is in your hand, can not be played.

At the beginning of the game, Bie-Bot was lucky to have drawn the only one *Grand Finale* in his starting hand, and he also knows that each card in the draw pile from top to bottom. Now, his goal is to successfully play the *Grand Finale* card. Bie-Bot wants to know, under his optimal strategy, what is the minimum hand size limit k required for him to achieve his goal. As the third smartest player in Pigeland, can you help him out?

More formally, you are given a string S_H of length n representing Bie-Bot's starting hand and a string S_P of length m representing the draw pile from top to bottom. Both strings consist of uppercase letters 'G', 'Q', 'B' and 'W', indicating that the card at the corresponding position in the starting hand or draw pile is *Grand Finale*, *Quick Slash*, *Backflip*, or *Wound*, respectively. Bie-Bot can use the cards in his hand according to the rules mentioned above. Please output the minimum hand size limit k ($k \geq n$) such that Bie-Bot can finally play the *Grand Finale* or state that there is no such k.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \leq n, m \leq 2500$) indicating the number of cards in Bie-Bot's starting hand and draw pile.

The second line contains a string S_H of length n to represent Bie-Bot's starting hand, consisting of uppercase letters 'G', 'Q', 'B', and 'W'. It's guaranteed that there is exactly one 'G' in S_H.

The third line of the input contains a string S_P of length m to represent Bie-Bot's draw pile from top to bottom, consisting of uppercase letters 'Q', 'B', and 'W'.

It is guaranteed that the sum of $(n + m)$ of all test cases will not exceed 5×10^4.

Output

For each test case output one line containing one integer indicating the minimum hand size k ($k \geq n$) that can lead to successfully playing *Grand Finale*, or `IMPOSSIBLE` if it can't be played.

Example

standard input	standard output
2 2 6 BG BQWBWW 4 6 GQBW WWWWQB	3 IMPOSSIBLE

Note

We express the situation with "hand/deck" string. For the first sample test case, one optimal strategy is:

1. BG/BQWBWW $\xrightarrow{B}$ BQG/WBWW
2. BQG/WBWW $\xrightarrow{Q}$ BWG/BWW
3. BWG/BWW $\xrightarrow{B}$ BWG/W (Discard one 'W' here due to hand size limit)

4. BWG/W $\xrightarrow{B}$ WWG/$\emptyset$ (Only draw one card here because there is no more card in the draw pile)
5. WWG/$\emptyset$ $\xrightarrow{G}$ Successfully playing *Grand Finale*!

Problem L. Elevator

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

There are n groups of parcels waiting to be delivered. There are c_i parcels in the i-th group, where each parcel has the same weight of w_i (w_i is either 1 or 2) and should be delivered to the f_i-th floor.

There is an elevator which can carry parcels with a maximum total weight of k (k is even) for each ride. The elevator will start from the ground floor and move gradually towards the hightest floor h where a parcel in the elevator should be delivered and finally move back to the ground floor, costing h units of electric power for that ride.

More formally, let (w, f) be a parcel whose weight is w and should be delivered to the f-th floor. A multiset (a set which allows duplicated elements) of parcels $\mathbb{P}$ can be delivered in the same ride if $\sum\limits_{(w,f)\in\mathbb{P}} w \le k$. This ride will cost $\max\limits_{(w,f)\in\mathbb{P}} f$ units of electric power.

What's the minimum total units of electric power needed to deliver all parcels?

Note that each ride can contain parcels from different groups and each group of parcels can be delivered through multiple rides. You can treat this problem as if there are a total of $\sum\limits_{i=1}^{n} c_i$ parcels to be delivered, just that some parcels may have the same weight and the same destination.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and k ($1 \le n \le 10^5$, $2 \le k \le 2 \times 10^{10}$, k is even) indicating the number of groups and the capacity of the elevator.

For the following n lines, the i-th line contains three integers c_i, w_i and f_i ($1 \le c_i \le 10^5$, $w_i \in \{1, 2\}$, $1 \le f_i \le 10^5$) indicating the number of parcels in the i-th group, the weight of each parcel and the destination of the parcels.

It's guaranteed that the sum of n over all test cases does not exceed 3×10^5.

Output

For each test case output one line containing one integer indicating the minimum total units of electric power needed to deliver all parcels.

Example

standard input	standard output
2 4 6 1 1 8 7 2 5 1 1 7 3 2 6 8 1200000 100000 1 100000 100000 1 12345 100000 2 100000 100000 2 12345 100000 1 100000 100000 1 12345 100000 2 100000 100000 2 12345	24 100000

Note

For the first sample test case we can follow this strategy:

- In the first ride, deliver parcel $(2, 6)$, $(2, 6)$ and $(2, 5)$. This ride costs 6 units of electric power.
- In the second ride, deliver parcel $(1, 8)$, $(1, 7)$, $(2, 6)$ and $(2, 5)$. This ride costs 8 units of electric power.
- In the third ride, deliver parcel $(2, 5)$, $(2, 5)$ and $(2, 5)$. This ride costs 5 units of electric power.
- In the fourth ride, deliver parcel $(2, 5)$ and $(2, 5)$. This ride costs 5 units of electric power.

The total units of electric power is $6 + 8 + 5 + 5 = 24$. It can be proven that this is the minimum total units of electric power needed.

For the second sample test case, all parcels can be delivered in the same ride.

Problem M. Trapping Rain Water

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

There is a histogram represented by an integer sequence $a_1, a_2, \cdots, a_n$ of length n. For the i-th bar from left to right, its height is a_i and its width is 1.

We'll perform q modifications to the histogram. The i-th modification can be represented by a pair of integers (x_i, v_i) indicating that we'll increase the height of the x_i-th bar by v_i.

After each modification, answer the following query: Calculate how much water this histogram can trap if a heavy rain pours onto it and fills all the pits as much as possible.

More formally, given an integer sequence $a_1, a_2, \cdots, a_n$ of length n, the i-th modification will increase a_{x_i} by v_i. After each modification, answer the following query: Let $f_i = \max(a_1, a_2, \cdots, a_i)$ and $g_i = \max(a_i, a_{i+1}, \cdots, a_n)$, calculate

$$\sum_{i=1}^{n} (\min(f_i, g_i) - a_i)$$

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

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

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($1 \le a_i \le 10^6$) where a_i indicates the initial height of the i-th bar.

The third line contains an integer q ($1 \le q \le 10^5$) indicating the number of modifications.

For the following q lines, the i-th line contains two integers x_i and v_i ($1 \le x_i \le n$, $1 \le v_i \le 10^6$) indicating that the i-th modification increases the height of the x_i-th bar by v_i.

It is guaranteed that neither the sum of n nor the sum of q of all test cases will exceed 10^6.

Output

For each modification output one line containing one integer indicating how much rain water this histogram can trap.

Example

standard input	standard output
2 6 1 2 3 4 5 6 2 1 2 3 3 5 100 10 1 10 100 1 3 100	1 4 180