A. Sum of 2050

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

A number is called *2050-number* if it is $2050, 20500, ..., (2050 \cdot 10^k$ for integer $k \geq 0$).

Given a number n, you are asked to represent n as the sum of some (not necessarily distinct) 2050-numbers. Compute the minimum number of 2050-numbers required for that.

Input

The first line contains a single integer T ($1 \leq T \leq 1\,000$) denoting the number of test cases.

The only line of each test case contains a single integer n ($1 \leq n \leq 10^{18}$) denoting the number to be represented.

Output

For each test case, output the minimum number of 2050-numbers in one line.

If n cannot be represented as the sum of 2050-numbers, output -1 instead.

Example

input
6 205 2050 4100 20500 22550 25308639900

output
-1 1 2 1 2 36

Note

In the third case, $4100 = 2050 + 2050$.

In the fifth case, $22550 = 20500 + 2050$.

B. Morning Jogging

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

The 2050 volunteers are organizing the "Run! Chase the Rising Sun" activity. Starting on Apr 25 at 7:30 am, runners will complete the 6km trail around the Yunqi town.

There are $n + 1$ checkpoints on the trail. They are numbered by $0, 1, ..., n$. A runner must start at checkpoint 0 and finish at checkpoint n. No checkpoint is skippable — he must run from checkpoint 0 to checkpoint 1, then from checkpoint 1 to checkpoint 2 and so on. Look at the picture in notes section for clarification.

Between any two adjacent checkpoints, there are m different paths to choose. For any $1 \leq i \leq n$, to run from checkpoint $i - 1$ to checkpoint i, a runner can choose exactly one from the m possible paths. The length of the j-th path between checkpoint $i - 1$ and i is $b_{i,j}$ for any $1 \leq j \leq m$ and $1 \leq i \leq n$.

To test the trail, we have m runners. Each runner must run from the checkpoint 0 to the checkpoint n once, visiting all the checkpoints. Every path between every pair of adjacent checkpoints needs to be ran by **exactly one** runner. If a runner chooses the path of length l_i between checkpoint $i - 1$ and i ($1 \leq i \leq n$), his *tiredness* is

$$\min_{i=1}^{n} l_i,$$

i. e. the minimum length of the paths he takes.

Please arrange the paths of the m runners to minimize the sum of tiredness of them.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10\,000$). Description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \leq n, m \leq 100$).

The i-th of the next n lines contains m integers $b_{i,1}$, $b_{i,2}$, ..., $b_{i,m}$ ($1 \leq b_{i,j} \leq 10^9$).

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

Output

For each test case, output n lines. The j-th number in the i-th line should contain the **length** of the path that runner j chooses to run from checkpoint $i - 1$ to checkpoint i. There should be exactly m integers in the i-th line and these integers should form a permuatation of $b_{i,1}$, ..., $b_{i,m}$ for all $1 \leq i \leq n$.
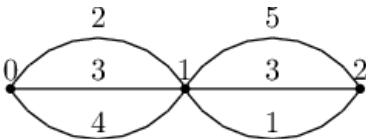
If there are multiple answers, print any.

Example

input
2
2 3
2 3 4
1 3 5
3 2
2 3
4 1
3 5

output
2 3 4
5 3 1
2 3
4 1
3 5

Note

In the first case, the sum of tiredness is $\min(2, 5) + \min(3, 3) + \min(4, 1) = 6$.

In the second case, the sum of tiredness is $\min(2, 4, 3) + \min(3, 1, 5) = 3$.

C. Fillomino 2

Fillomino is a classic logic puzzle. (You do not need to know Fillomino in order to solve this problem.) In one classroom in Yunqi town, some volunteers are playing a board game variant of it:

Consider an n by n chessboard. Its rows are numbered from 1 to n from the top to the bottom. Its columns are numbered from 1 to n from the left to the right. A cell on an intersection of x-th row and y-th column is denoted (x, y). The main diagonal of the chessboard is cells (x, x) for all $1 \leq x \leq n$.

A permutation of $\{1, 2, 3, \ldots, n\}$ is written on the main diagonal of the chessboard. There is exactly one number written on each of the cells. The problem is to partition the cells under and on the main diagonal (there are exactly $1 + 2 + \ldots + n$ such cells) into n connected regions satisfying the following constraints:

1. Every region should be connected. That means that we can move from any cell of a region to any other cell of the same region visiting only cells of the same region and moving from a cell to an adjacent cell.
2. The x-th region should contain cell on the main diagonal with number x for all $1 \leq x \leq n$.
3. The number of cells that belong to the x-th region should be equal to x for all $1 \leq x \leq n$.
4. Each cell under and on the main diagonal should belong to exactly one region.

Input

The first line contains a single integer n ($1 \leq n \leq 500$) denoting the size of the chessboard.

The second line contains n integers p_1, p_2, ..., p_n. p_i is the number written on cell (i, i). It is guaranteed that each integer from $\{1, \ldots, n\}$ appears exactly once in p_1, ..., p_n.

Output

If no solution exists, output -1.

Otherwise, output n lines. The i-th line should contain i numbers. The j-th number on the i-th line should be x if cell (i, j) belongs to the **the region with x cells**.
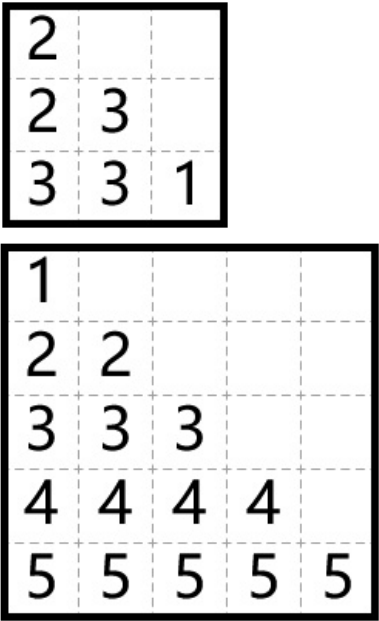
Examples

input
3 2 3 1

output
2 2 3 3 3 1

input
5 1 2 3 4 5

output
1 2 2 3 3 3 4 4 4 4 5 5 5 5 5

Note

The solutions to the examples are illustrated in the following pictures:

D. Explorer Space

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

You are wandering in the explorer space of the 2050 Conference.

The explorer space can be viewed as an undirected weighted grid graph with size $n \times m$. The set of vertices is $\{(i, j) | 1 \le i \le n, 1 \le j \le m\}$. Two vertices (i_1, j_1) and (i_2, j_2) are connected by an edge if and only if $|i_1 - i_2| + |j_1 - j_2| = 1$.

At each step, you can walk to any vertex connected by an edge with your current vertex. On each edge, there are some number of exhibits. Since you already know all the exhibits, whenever you go through an edge containing x exhibits, your *boredness* increases by x.

For each starting vertex (i, j), please answer the following question: What is the minimum possible boredness if you walk from (i, j) and go back to it after exactly k steps?

You can use any edge for multiple times but the boredness on those edges are also counted for multiple times. At each step, you cannot stay on your current vertex. You also cannot change direction while going through an edge. Before going back to your starting vertex (i, j) after k steps, you can visit (i, j) (or not) freely.

Input

The first line contains three integers n, m and k ($2 \le n, m \le 500, 1 \le k \le 20$).

The j-th number ($1 \le j \le m-1$) in the i-th line of the following n lines is the number of exibits on the edge between vertex (i, j) and vertex $(i, j+1)$.

The j-th number ($1 \le j \le m$) in the i-th line of the following $n-1$ lines is the number of exibits on the edge between vertex (i, j) and vertex $(i+1, j)$.

The number of exhibits on each edge is an integer between 1 and 10^6.

Output

Output n lines with m numbers each. The j-th number in the i-th line, $answer_{ij}$, should be the minimum possible boredness if you walk from (i, j) and go back to it after exactly k steps.

If you cannot go back to vertex (i, j) after exactly k steps, $answer_{ij}$ should be -1.

Examples

input
3 3 10
1 1
1 1
1 1
1 1 1
1 1 1

output
10 10 10
10 10 10
10 10 10

input
2 2 4
1
3
4 2

output
4 4
10 6

input
2 2 3
1
2
3 4

output
-1 -1
-1 -1

Note

In the first example, the answer is always 10 no matter how you walk.

In the second example, $answer_{21} = 10$, the path is $(2, 1) \rightarrow (1, 1) \rightarrow (1, 2) \rightarrow (2, 2) \rightarrow (2, 1)$, the boredness is $4 + 1 + 2 + 3 = 10$.

E. Group Photo

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

In the 2050 Conference, some people from the competitive programming community meet together and are going to take a photo. The n people form a line. They are numbered from 1 to n from left to right. Each of them either holds a cardboard with the letter 'C' or a cardboard with the letter 'P'.

Let $C = \{c_1, c_2, \ldots, c_m\}$ ($c_1 < c_2 < \ldots < c_m$) be the set of people who hold cardboards of 'C'. Let $P = \{p_1, p_2, \ldots, p_k\}$ ($p_1 < p_2 < \ldots < p_k$) be the set of people who hold cardboards of 'P'. The photo is good if and only if it satisfies the following constraints:

1. $C \cup P = \{1, 2, \ldots, n\}$
2. $C \cap P = \emptyset$.
3. $c_i - c_{i-1} \le c_{i+1} - c_i \, (1 < i < m)$.
4. $p_i - p_{i-1} \ge p_{i+1} - p_i \, (1 < i < k)$.

Given an array $a_1, \ldots, a_n$, please find the number of good photos satisfying the following condition:

$$\sum_{x \in C} a_x < \sum_{y \in P} a_y.$$

The answer can be large, so output it modulo $998\,244\,353$. Two photos are different if and only if there exists at least one person who holds a cardboard of 'C' in one photo but holds a cardboard of 'P' in the other.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 200\,000$). Description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 200\,000$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$).

It is guaranteed that the sum of n over all test cases does not exceed $200\,000$.

Output

For each test case, output the answer modulo $998\,244\,353$ in a separate line.

Example

input
3
5
2 1 2 1 1
4
9 2 2 2
1
998244353

output
10
7
1

Note

For the first test case, there are 10 possible good photos satisfying the condition: PPPPP, CPPPP, PCPPP, CCPPP, PCCPP, PCPCP, PPPPC, CPPPC, PCPPC, PPPCC.

For the second test case, there are 7 possible good photos satisfying the condition: PPPP, PCPP, PCCP, PPPC, PCPC, PPCC, PCCC.

F. Reunion

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

It is reported that the 2050 Conference will be held in Yunqi Town in Hangzhou from April 23 to 25, including theme forums, morning jogging, camping and so on.

The relationship between the n volunteers of the 2050 Conference can be represented by a tree (a connected undirected graph with n vertices and $n - 1$ edges). The n vertices of the tree corresponds to the n volunteers and are numbered by $1, 2, \ldots, n$.

We define the distance between two volunteers i and j, $\mathrm{dis}(i, j)$ as the number of edges on the shortest path from vertex i to vertex j on the tree. $\mathrm{dis}(i, j) = 0$ whenever $i = j$.

Some of the volunteers can attend the on-site reunion while others cannot. If for some volunteer x and nonnegative integer r, all volunteers whose distance to x is no more than r can attend the on-site reunion, a forum with radius r can take place. The *level* of the on-site reunion is defined as the maximum possible radius of any forum that can take place.

Assume that each volunteer can attend the on-site reunion with probability $\frac{1}{2}$ and these events are independent. Output the expected level of the on-site reunion. When no volunteer can attend, the level is defined as -1. When all volunteers can attend, the level is defined as n.

Input

The first line contains a single integer n ($2 \le n \le 300$) denoting the number of volunteers.

Each of the next $n - 1$ lines contains two integers a and b denoting an edge between vertex a and vertex b.

Output

Output the expected level modulo $998\,244\,353$.

Formally, let $M = 998\,244\,353$. It can be shown that the answer can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output such an integer x that $0 \le x < M$ and $x \cdot q \equiv p \pmod{M}$.

Examples

input
3 1 2 2 3
output
499122177

input
5 1 2 2 3 3 4 3 5
output
249561089

input
10 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10
output
821796866

Note

For the first example, the following table shows all possible outcomes. yes means the volunteer can attend the on-site reunion and no means he cannot attend.

1	2	3	$level$
yes	yes	yes	3
yes	yes	no	1
yes	no	yes	0
yes	no	no	0
no	yes	yes	1
no	yes	no	0
no	no	yes	0
no	no	no	-1

The expected level is $\frac{3+1+1+(-1)}{2^3} = \frac{1}{2}$.

G. Starry Night Camping

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

At the foot of Liyushan Mountain, n tents will be carefully arranged to provide accommodation for those who are willing to experience the joy of approaching nature, the tranquility of the night, and the bright starry sky.

The i-th tent is located at the point of (x_i, y_i) and has a weight of w_i. A tent is *important* if and only if both x_i and y_i are even. You need to remove some tents such that for each remaining important tent (x, y), there do not exist 3 other tents (x'_1, y'_1), (x'_2, y'_2) and (x'_3, y'_3) such that both conditions are true:

1. $|x'_j - x|, |y'_j - y| \leq 1$ for all $j \in \{1, 2, 3\}$, and
2. these four tents form a parallelogram (or a rectangle) and one of its sides is **parallel to the x-axis**.

Please maximize the sum of the weights of the tents that are **not** removed. Print the maximum value.

Input

The first line contains a single integer n ($1 \leq n \leq 1\,000$), representing the number of tents.

Each of the next n lines contains three integers x_i, y_i and w_i ($-10^9 \leq x_i, y_i \leq 10^9$, $1 \leq w_i \leq 10^9$), representing the coordinate of the i-th tent and its weight. No two tents are located at the same point.

Output

A single integer — the maximum sum of the weights of the remaining tents.
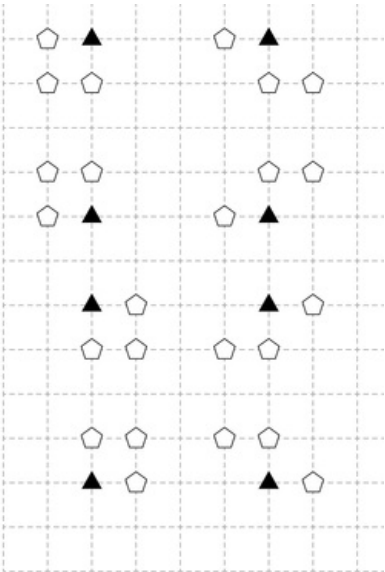
Examples

input
5 0 0 4 0 1 5 1 0 3 1 1 1 -1 1 2
output
12

input
32 2 2 1 2 3 1 3 2 1 3 3 1 2 6 1 2 5 1 3 6 1 3 5 1 2 8 1 2 9 1 1 8 1 1 9 1 2 12 1 2 11 1 1 12 1 1 11 1 6 2 1 7 2 1 6 3 1 5 3 1 6 6 1 7 6 1 5 5 1 6 5 1 6 8 1 5 8 1 6 9 1 7 9 1 6 12 1 5 12 1 6 11 1 7 11 1
output
24

Note

Here is an illustration of the second example. Black triangles indicate the important tents. This example also indicates all 8 forbidden patterns.

H. Fly Around the World

time limit per test: 5 seconds

memory limit per test: 256 megabytes

After hearing the story of Dr. Zhang, Wowo decides to plan his own flight around the world.

He already chose n checkpoints in the world map. Due to the landform and the clouds, he cannot fly too high or too low. Formally, let b_i be the height of Wowo's aircraft at checkpoint i, $x_i^- \le b_i \le x_i^+$ should be satisfied for all integers i between 1 and n, where x_i^- and x_i^+ are given integers.

The angle of Wowo's aircraft is also limited. For example, it cannot make a 90-degree climb. Formally, $y_i^- \le b_i - b_{i-1} \le y_i^+$ should be satisfied for all integers i between 2 and n, where y_i^- and y_i^+ are given integers.

The final limitation is the speed of angling up or angling down. An aircraft should change its angle slowly for safety concerns. Formally, $z_i^- \le (b_i - b_{i-1}) - (b_{i-1} - b_{i-2}) \le z_i^+$ should be satisfied for all integers i between 3 and n, where z_i^- and z_i^+ are given integers.

Taking all these into consideration, Wowo finds that the heights at checkpoints are too hard for him to choose. Please help Wowo decide whether there exists a sequence of **real** numbers $b_1, \ldots, b_n$ satisfying all the contraints above.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 66\,666$). Description of the test cases follows.

The first line of each test case contains a single integer n ($3 \le n \le 100\,000$).

The i-th of the next n lines contains two integers x_i^-, x_i^+ ($-10^8 \le x_i^- \le x_i^+ \le 10^8$) denoting the lower and upper bound of b_i.

The i-th of the next $n-1$ lines contains two integers y_{i+1}^-, y_{i+1}^+ ($-10^8 \le y_{i+1}^- \le y_{i+1}^+ \le 10^8$) denoting the lower and upper bound of $b_{i+1} - b_i$.

The i-th of the next $n-2$ lines contains two integers z_{i+2}^-, z_{i+2}^+ ($-10^8 \le z_{i+2}^- \le z_{i+2}^+ \le 10^8$) denoting the lower and upper bound of $(b_{i+2} - b_{i+1}) - (b_{i+1} - b_i)$.

It is guaranteed that the sum of n over all test cases does not exceed $200\,000$.

It is guaranteed that relaxing every constraint by 10^{-6} (i.e., decrease x_i^-, y_i^-, z_i^- by 10^{-6} and increase x_i^+, y_i^+, z_i^+ by 10^{-6}) will not change the answer.

Output

For each test case, output YES if a sequence $b_1, \ldots, b_n$ satisfying the constraints exists and NO otherwise. The sequence $b_1, \ldots, b_n$ is **not** required.

Example

input
4
3
0 1
0 1
0 1
1 1
1 1
-100 100
3
-967 541
-500 834
-724 669
-858 978
-964 962
-645 705
4
0 0
0 1
0 1
1 1
0 1
0 1
0 1
0 0
0 0
4
0 0
33 34
65 66
100 100
0 100
0 100
0 100
0 0
0 0

output
NO
YES

```
YES
NO
```

Note

In the first test case, all b_i's are in $[0, 1]$. Because of the constraints $1 = y_2^- \leq b_2 - b_1 \leq y_2^+ = 1$, $b_2 - b_1$ must be 1. So $b_2 = 1$ and $b_1 = 0$ must hold. Then by $1 = y_3^- \leq b_3 - b_2 \leq y_3^+ = 1$, b_3 equals 2. This contradicts the constraint of $b_3 \leq 1$. So no solution exists.

In the second test case, we can let all b_i's be 0.

In the third test case, one possible solution is $b_1 = 0$, $b_2 = 1/3$, $b_3 = 2/3$, $b_4 = 1$.