Problem A. Oops, It's Yesterday Twice More

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

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

Team ***Power of Two*** and team ***Three Hold Two*** won the champion for Tsinghua University in 2018 and 2019. In 2020, team ***Inverted Cross*** from Peking University won the champion. In 2021, there are around 700 teams including **the defending champion** participating in the contest. We are so excited to see who will win this year!

Although we can't gather in Nanjing this time due to the pandemic, we should still 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!

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;
}
```

Now in the 2021 contest, Paimon prepares another version of the problem for you. You are given a grid with n rows and n columns ($2 \le n \le 500$). All cells are empty and there is one kangaroo standing in each cell.

Similarly, you 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. 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 $(i - 1, j)$ if $i > 1$. Otherwise, it will stay in the same grid.

2. Button D: it will move to $(i+1, j)$ if $i < n$. Otherwise, it will stay in the same grid.
3. Button L: it will move to $(i, j-1)$ if $j > 1$. Otherwise, it will stay in the same grid.
4. Button R: it will move to $(i, j+1)$ if $j < n$. Otherwise, it will stay in the same grid.

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 $3(n-1)$.

Input

There is only one test case in each test file.

The first and only line of the input contains three integers n, a, b ($2 \le n \le 500$, $1 \le a, b \le n$) indicating the size of the grid and the target cell.

Output

Output a string consisting only of characters 'U', 'D', 'L' and 'R' in one line. And its length mustn't exceed $3(n-1)$. It can be proved that the answer always exists.

Examples

standard input	standard output
3 3 3	RRDD
4 3 2	DLDLDLUR

Problem B. Puzzle in Inazuma

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

Every traveler knows that they'll be rewarded with a treasure box after solving the puzzles in Inazuma, but few know that these puzzles are designed by Yae Miko, the Guuji of the Grand Narukami Shrine, to test whether the traveler is strong enough to save her friend Raiden Shogun and people of Inazuma.

Grand Narukami Shrine

After a traveler passes the test Yae will have to reset the puzzles to the initial state. But this time she has some troubles and even doubts that whether some of them are already broken.

Yae's puzzle can be considered as a weighted undirected complete graph G before resetting. We also denote the initial state as another weighted undirected complete graph H. Both G and H have exactly n vertices, and these vertices are labeled from 1 to n.

To reset graph G to H Yae can perform the following operation any number of times:

- First select four distinct vertices a, b, c, d and an integer x. Note that she can select a different set of a, b, c, d and x each time.
- Let (i, j) be the edge between vertices i and j. Increase the weight of (a, b), (a, c) and (a, d) by x and also decrease the weight of (b, c), (b, d) and (c, d) by x.

Please help Yae determine whether she can change graph G to graph H. If yes you also shall tell her the detailed steps.

Input

There is only one test case in each test file.

The first line of the input contains an integer n ($4 \le n \le 100$) indicating the number of vertices in graph G and H.

For the following $(n - 1)$ lines, the i-th line contains $(n - i)$ integers $w_{i,i+1}, w_{i,i+2}, \cdots, w_{i,n}$ ($-100 \le w_{i,j} \le 100$) where $w_{i,j}$ indicates the weight of the edge connecting vertices i and j in graph G.

For the following $(n-1)$ lines, the i-th line contains $(n-i)$ integers $v_{i,i+1}, v_{i,i+2}, \cdots, v_{i,n}$ $(-100 \le v_{i,j} \le 100)$ where $v_{i,j}$ indicates the weight of the edge connecting vertices i and j in graph H.

Output

If Yae cannot change G to H output "-1" (without quotes).

Otherwise first output an integer m $(0 \le m \le 10^5)$ in one line indicating the number of operations Yae needs.

For the following m lines, output five integers a_i, b_i, c_i, d_i and x_i in the i-th line separated by a space, indicating that for the i-th operation Yae choose vertices a_i, b_i, c_i, d_i and integer x_i. Note that a_i, b_i, c_i, d_i must be distinct and $-10^9 \le x_i \le 10^9$.

It can be proved that if graph G can be changed to graph H there exists a solution with no more than 10^5 operations.

Note that you don't have to minimize m. If there are multiple solutions, output any of them.

Examples

standard input	standard output
4 0 1 1 0 0 1 1 0 0 1 1 0	1 2 1 3 4 1
4 3 3 3 0 0 0 0 0 0 3 3 3	1 1 2 3 4 -3
5 -12 15 -12 1 37 14 7 7 9 -11 12 5 1 13 -1 -4 -7 -5 -9 18	-1

Problem C. Klee in Solitary Confinement

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

Since the traveler comes, People in Monstadt suddenly raise great interest in computer programming and algorithms, including Klee, the Spark Knight of the Knights of Favonius.

Source: Genshin Impact Official

Being sent to solitary confinement by Jean again, Klee decides to spend time learning the famous Mo's algorithm, which can compute with a time complexity of $\mathcal{O}(n^{1.5})$ for some range query problem without modifications.

To check whether Klee has truly mastered the algorithm (or in fact making another bombs secretly), Jean gives her a problem of an integer sequence $a_1, a_2, \cdots, a_n$ along with some queries $[l_i, r_i]$ requiring her to find the mode number in the contiguous subsequence $a_{l_i}, a_{l_i+1}, \cdots, a_{r_i}$. The mode number is the most common number (that is to say, the number which appears the maximum number of times) in the subsequence.

With the help of Mo's algorithm, Klee solves that problem without effort, but another problem comes into her mind. Given an integer sequence $a_1, a_2, \cdots, a_n$ of length n and an integer k, you can perform the following operation at most once: Choose two integers l and r such that $1 \le l \le r \le n$ and add k to every a_i where $l \le i \le r$. Note that it is OK not to perform this operation. Compute the maximum occurrence of the mode number of the whole sequence if you choose to perform (or not perform) the operation optimally.

Input

There is only one test case in each test file.

The first line of the input contains two integers n and k ($1 \le n \le 10^6$, $-10^6 \le k \le 10^6$) indicating the length of the sequence and the additive number.

The second line of the input contains n integers $a_1, a_2, \cdots, a_n$ ($-10^6 \le a_i \le 10^6$) indicating the original sequence.

Output

Output one line containing one integer indicating the maximum occurrence of the mode number of the whole sequence after performing (or not performing) the operation.

Examples

standard input	standard output
5 2 2 2 4 4 4	5
7 1 3 2 3 2 2 2 3	6
7 1 2 3 2 3 2 3 3	5
9 -100 -1 -2 1 2 -1 -2 1 -2 1	3

Note

For the first sample test case, choose $l = 1$ and $r = 2$ and we'll result in the sequence $\{4, 4, 4, 4, 4\}$. The mode number is obviously 4 which appears 5 times.

For the second sample test case, choose $l = 4$ and $r = 6$ and we'll result in the sequence $\{3, 2, 3, 3, 3, 3, 3\}$. The mode number is 3 which appears 6 times.

For the fourth sample test case, choose not to perform the operation. The mode number is 1 and -2 which both appear 3 times.

Problem D. Paimon Sorting

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

Paimon just invents a new sorting algorithm which looks much like *bubble sort*, with a few differences. It accepts a 1-indexed sequence A of length n and sorts it. Its pseudo-code is shown below.

Functions 1 The Sorting Algorithm

```
1: function SORT(A)
2:     for i ← 1 to n do          ▷ n is the number of elements in A
3:         for j ← 1 to n do
4:             if a_i < a_j then          ▷ a_i is the i-th element in A
5:                 Swap a_i and a_j
```

If you don't believe this piece of algorithm can sort a sequence it will also be your task to prove it. Anyway here comes the question:

Given an integer sequence $A = a_1, a_2, \cdots, a_n$ of length n, for each of its prefix A_k of length k (that is, for each $1 \le k \le n$, consider the subsequence $A_k = a_1, a_2, \cdots, a_k$), count the number of swaps performed if we call SORT(A_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 an integer n ($1 \le n \le 10^5$) indicating the length of the sequence.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($1 \le a_i \le n$) indicating the given sequence.

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 $s_1, s_2, \cdots, s_n$ separated by a space, where s_i is the number of swaps performed if we call SORT(A_i).

Example

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

Problem E. Paimon Segment Tree

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

Paimon just learns the persistent segment tree and decides to practice immediately. Therefore, Lumine gives her an easy problem to start:

Given a sequence $a_1, a_2, \cdots, a_n$ of length n, Lumine will apply m modifications to the sequence. In the i-th modification, indicated by three integers l_i, r_i ($1 \le l_i \le r_i \le n$) and x_i, Lumine will change a_k to $(a_k + x_i)$ for all $l_i \le k \le r_i$.

Let $a_{i,t}$ be the value of a_i just after the t-th operation. This way we can keep track of all historial versions of a_i. Note that $a_{i,t}$ might be the same as $a_{i,t-1}$ if it hasn't been modified in the t-th modification. For completeness we also define $a_{i,0}$ as the initial value of a_i.

After all modifications have been applied, Lumine will give Paimon q queries about the sum of squares among the historical values. The k-th query is indicated by four integers l_k, r_k, x_k and y_k and requires Paimon to calculate

$$\sum_{i=l_k}^{r_k} \sum_{j=x_k}^{y_k} a_{i,j}^2$$

Please help Paimon compute the result for all queries. As the answer might be very large, please output the answer modulo $10^9 + 7$.

Input

There is only one test case in each test file.

The first line of the input contains three integers n, m and q ($1 \le n, m, q \le 5 \times 10^4$) indicating the length of the sequence, the number of modifications and the number of queries.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($|a_i| < 10^9 + 7$) indicating the initial sequence.

For the following m lines, the i-th line contains three integers l_i, r_i and x_i ($1 \le l_i \le r_i \le n$, $|x_i| < 10^9 + 7$) indicating the i-th modification.

For the following q lines, the i-th line contains four integers l_i, r_i, x_i and y_i ($1 \le l_i \le r_i \le n$, $0 \le x_i \le y_i \le m$) indicating the i-th query.

Output

For each query output one line containing one integer indicating the answer modulo $10^9 + 7$.

Examples

standard input	standard output
3 1 1 8 1 6 2 3 2 2 2 0 0	1
4 3 3 2 3 2 2 1 1 6 1 3 3 1 3 6 2 2 2 3 1 4 1 3 4 4 2 3	180 825 8

Problem F. Paimon Polygon

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

Paimon just puts $(n+1)$ distinct points on the plane, one of which is a special point $O=(0,0)$, and denote the group of remaining points as $\mathbb{S}$.

We call a point set $\mathbb{U}$ *strict convex set*, if and only if $|\mathbb{U}| \geq 3$ and all the points from $\mathbb{U}$ lie exactly on the convex hull built from $\mathbb{U}$, with no three points lying on the same line.

You should divide $\mathbb{S}$ into two sets $\mathbb{A}$ and $\mathbb{B}$ so that:

- $\mathbb{A} \cap \mathbb{B} = \emptyset$.
- $\mathbb{A} \cup \mathbb{B} = \mathbb{S}$.
- $|\mathbb{A}| \geq 2, |\mathbb{B}| \geq 2$.
- The point set $\mathbb{A} \cup \{O\}$ is a *strict convex set*, and denote its convex hull as $C_{\mathbb{A}\cup\{O\}}$.
- The point set $\mathbb{B} \cup \{O\}$ is a *strict convex set*, and denote its convex hull as $C_{\mathbb{B}\cup\{O\}}$.
- The outlines(edges) of $C_{\mathbb{A}\cup\{O\}}$ and $C_{\mathbb{B}\cup\{O\}}$ only intersect at point O. That is, only one point O satisfies that it lies both on the outlines of $C_{\mathbb{A}\cup\{O\}}$ and $C_{\mathbb{B}\cup\{O\}}$.

Please help Paimon to maximize the sum of the perimeters of these two convex hulls. That is, find a valid division $\mathbb{A}$ and $\mathbb{B}$ which maximizes $(L(C_{\mathbb{A}\cup\{O\}}) + L(C_{\mathbb{B}\cup\{O\}}))$, where $L(\text{polygon})$ means the perimeter of that polygon.

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 one integer n $(4 \leq n \leq 5 \times 10^5)$ indicating the number of points in $\mathbb{S}$.

For the following n lines, the i-th line contains two integers x_i and y_i $(-10^9 \leq x_i, y_i \leq 10^9, (x_i, y_i) \neq (0,0))$ indicating the location of the i-th point in $\mathbb{S}$.

It's guaranteed that the points given in the same test case are pairwise different. However, there may be three points lying on the same line.

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

Output

For each test case output one line containing a number indicating the maximum total perimeter. If there does not exist a valid division output "0" (without quotes) instead.

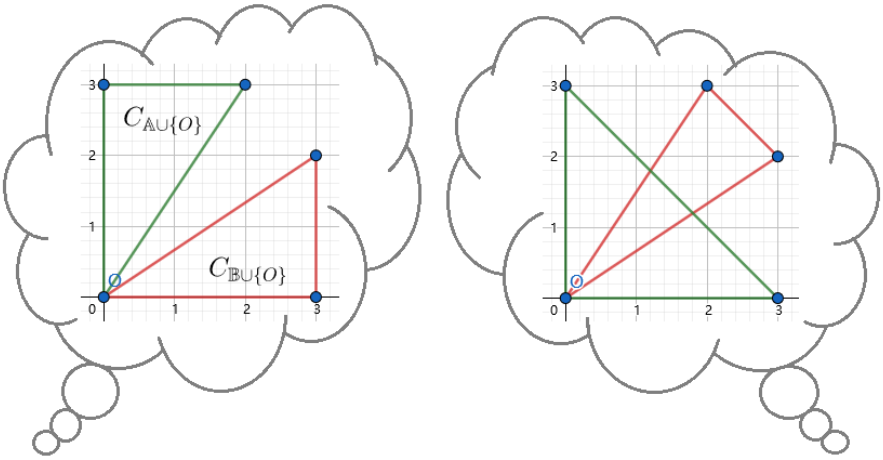
Your answer will be accepted if the relative or absolute error is less than 10^{-6}.

Example

standard input	standard output
3 4 0 3 3 0 2 3 3 2 5 4 0 5 -5 -4 -2 1 -2 -5 -2 4 0 1 1 0 0 2 1 1	17.2111025509 36.6326947621 0.0000000000

Note

A valid division (left) and an invalid division (right) of the first sample test case are shown below.



Problem G. Paimon's Tree

Input file:	standard input
Output file:	standard output
Time limit:	4 seconds
Memory limit:	256 megabytes

Paimon has found a tree with $(n+1)$ initially white vertices in her left pocket and decides to play with it. A tree with $(n+1)$ nodes is an undirected connected graph with n edges.

Paimon will give you an integer sequence $a_1, a_2, \cdots, a_n$ of length n. We first need to select a vertex in the tree and paint it black. Then we perform the following operation n times.

During the i-th operation, we select a white vertex x_i which is directly connected with a black vertex y_i by an edge, set the weight of that edge to a_i and also paint x_i in black. After these n operations we get a tree whose edges are all weighted.

What's the maximum length of the diameter of the weighted tree if we select the vertices optimally? The diameter of a weighted tree is the longest simple path in that tree. The length of a simple path is the sum of the weights of all edges in that path.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 5 \times 10^3$) indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 150$) indicating the length of the sequence.

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

For the following n lines, the i-th line contains two integers u_i and v_i ($1 \le u_i, v_i \le n+1$) indicating that there is an edge connecting vertex u_i and v_i in the tree.

It's guaranteed that there is at most 10 test cases satisfying $n > 20$.

Output

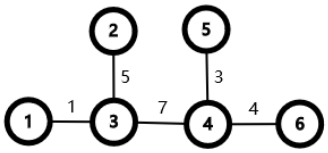
For each test case output one line containing one integer indicating the maximum length of the diameter of the tree.

Example

standard input	standard output
2 5 1 7 3 5 4 1 3 2 3 3 4 4 5 4 6 1 1000000000 1 2	16 1000000000

Note

For the first sample test case, we select the vertices in the order of $1, 3, 4, 5, 2, 6$, resulting in the weighted tree of the following image. It's obvious that the longest simple path is of length 16.


2
5
5
3
1
1
3
7
4
4
6

Problem H. Crystalfly

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

Paimon is catching crystalflies on a tree, which are a special kind of butterflies in Teyvat. A tree is a connected graph consisting of n vertices and $(n-1)$ undirected edges.

Pixiv ID: 93964680

There are initially a_i crystalflies on the i-th vertex. When Paimon reaches a vertex, she can catch all the remaining crystalflies on the vertex immediately. However, the crystalflies are timid. When Paimon reaches a vertex, all the crystalflies on the adjacent vertices will be disturbed. For the i-th vertex, if the crystalflies on the vertex are disturbed for the first time at the beginning of the t'-th second, they will disappear at the end of the $(t'+t_i)$-th second.

At the beginning of the 0-th second, Paimon reaches vertex 1 and stays there before the beginning of the 1-st second. Then at the beginning of each following second, she can choose one of the two operations:

- Move to one of the adjacent vertices of her current vertex and stay there before the beginning of the next second (if the crystalflies in the destination will disappear at the end of that second she can still catch them).
- Stay still in her current vertex before the beginning of the next second.

Calculate the maximum number of crystalflies Paimon can catch in $10^{10^{10^{10}}}$ seconds.

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 vertices.

The second line contains n integers $a_1, a_2, \cdots, a_n$ $(1 \le a_i \le 10^9)$ where a_i is the number of crystalflies on the i-th vertex.

The third line contains n integers $t_1, t_2, \cdots, t_n$ $(1 \le t_i \le 3)$ where t_i is the time before the crystalflies on the i-th vertex disappear after disturbed.

For the next $(n-1)$ lines, the i-th line contains two integers u_i and v_i $(1 \le u_i, v_i \le n)$ indicating an edge connecting vertices u_i and v_i in the tree.

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

Output

For each test case output one line containing one integer indicating the maximum number of crystalflies Paimon can catch.

Example

standard input	standard output
2 5 1 10 100 1000 10000 1 2 1 1 1 1 2 1 3 2 4 2 5 5 1 10 100 1000 10000 1 3 1 1 1 1 2 1 3 2 4 2 5	10101 10111

Note

For the first sample test case, follow the strategy below.

- During the 0-th second
  - Paimon arrives at vertex 1;
  - Paimon catches 1 crystalfly;
  - Crystalflies in vertices 2 and 3 are disturbed.
- During the 1-st second
  - Paimon arrives at vertex 3;
  - Paimon catches 100 crystalflies.
- During the 2-nd second
  - Paimon arrives at vertex 1;
  - Crystalflies in vertex 2 disappears.
- During the 3-rd second
  - Paimon arrives at vertex 2;
  - Crystalflies in vertices 4 and 5 are disturbed.
- During the 4-th second
  - Paimon arrives at vertex 5;
  - Paimon catches 10000 crystalflies;
  - Crystalflies in vertex 4 disappears.

For the second sample test case, the optimal strategy is the same with the first sample test case. Crystalflies in vertex 2 are scheduled to disappear at the end of the 3-rd (instead of the 2-nd) second, allowing Paimon to catch them.

Problem I. Cloud Retainer's Game

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

Cloud Retainer, the builder of the Dwelling in the clouds above Qingyun Peak, is very interested in mechanics. Although there is more than one month away from the Lantern Rite Festival in Liyue, she has already started the design of a gaming event for it.

Cloud Retainer and the Traveler

The game is mainly about releasing pinballs to get a score as high as possible. It is played on the 2-dimensional plane with two horizontal straight lines $y = 0$ and $y = H$. Between the two lines, there are n tiny wooden boards and m coins, both can be regarded as single points. The i-th wooden board is located at (x_i, y_i) while the i-th coin is located at (x'_i, y'_i).

A pinball is released from $(10^{-9}, 10^{-9})$ by the player. Let $\overrightarrow{v} = (v_x, v_y)$ be the velocity of the ball (that is to say, if the ball is currently located at (x, y) it will move to $(x + v_x\epsilon, y + v_y\epsilon)$ after ϵ seconds). Initially $\overrightarrow{v} = (1, 1)$.

When the ball hits a wooden board or one of the two horizontal straight lines, v_y will be negated (that is, v_y becomes $-v_y$) while v_x remains unchanged. If the ball hits a coin, the player's score is increased by 1 and the velocity of the ball remains unchanged.

To gain a higher score, the player can choose to remove any number of wooden boards before the pinball is released. It is also OK not to remove any wooden board. Cloud Retainer wants you to help her estimate the difficulty by computing the maximum score the player can get after $10^{10^{10^{10}}}$ seconds under the best strategy?

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 one integer H ($2 \le H \le 10^9$).

The second line contains one integer n ($1 \le n \le 10^5$) indicating the number of wooden boards.

For the following n lines, the i-th line contains two integers x_i and y_i ($1 \le x_i \le 10^9$, $1 \le y_i < H$) indicating a wooden board located at (x_i, y_i).

The following line contains one integer m ($1 \le m \le 10^5$) indicating the number of coins.

For the following m lines, the i-th line contains two integers x'_i and y'_i ($1 \le x'_i \le 10^9$, $1 \le y'_i < H$) indicating a coin located at (x'_i, y'_i).

It's guaranteed that the given $(n + m)$ points in the same test case will be distinct. It's also guaranteed that neither the sum of n nor the sum of m of all test cases will exceed 5×10^5.

Output

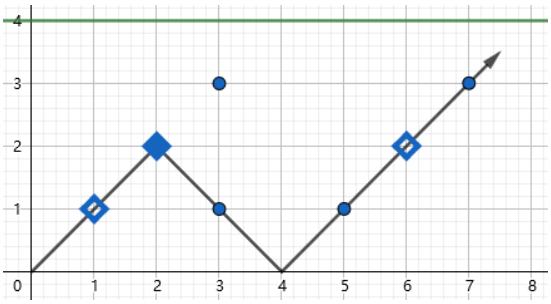
For each test case output one line containing one integer indicating the maximum score the player can get after removing some (or not removing any) wooden boards.

Example

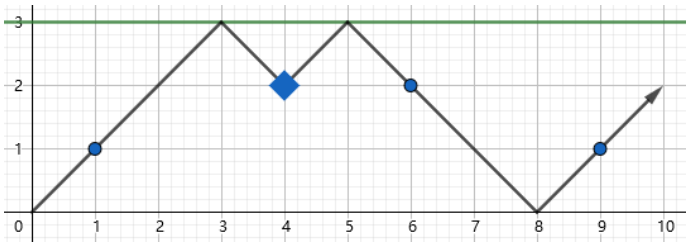
standard input	standard output
2 4 3 1 1 2 2 6 2 4 3 1 3 3 5 1 7 3 3 1 4 2 3 1 1 6 2 9 1	3 3

Note

The two sample test cases are shown below. Solid diamonds represent the remaining wooden boards, while hollow diamonds represent the removed wooden boards and round dots represent the coins.



Sample test case No. 1



Sample test case No. 2

Problem J. Xingqiu's Joke

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

Once again, Xingqiu hides Chongyun's ice cream into a box with a strange lock. Liyue's summer has been always very hot and Chongyun suffers more because of his excessive yang (positive) energy, so he needs that ice cream desperately.

Pixiv ID: 86787400

There are two integers a and b on the lock. Chongyun can perform the following three types of operations any number of times:

- Minus 1 from both a and b;
- Plus 1 to both a and b;
- Divide both a and b by one of their common **prime** factor (that is to say, divide them by a **prime** g where a and b are both divisible by g).

The box will be unlocked if either a or b or both become 1. To help Chongyun gets the ice cream back as quickly as possible, please tell him the minimum number of operations needed to unlock the box.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 300$) indicating the number of test cases. For each test case:

The first and only line contains two integers a and b ($1 \le a, b \le 10^9$, $a \ne b$).

Output

For each test case output one line containing one integer indicating the minimum number of operations to make a or b or both equal 1.

Example

standard input	standard output
5 4 7 9 8 32 84 11 35 2 1	2 7 5 4 0

Note

For the first sample test case, the optimal way is $(4, 7) \to (3, 6) \to (1, 2)$.

For the second sample test case, the optimal way is to apply the first type of operation 7 times.

For the third sample test case, the optimal way is $(32, 84) \to (16, 42) \to (15, 41) \to (14, 40) \to (13, 39) \to (1, 3)$.

For the fourth sample test case, the optimal way is $(11, 35) \to (12, 36) \to (6, 18) \to (2, 6) \to (1, 3)$.

Problem K. Ancient Magic Circle in Teyvat

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

Astrologist Mona Megistus discovers an ancient magic circle in Teyvat recently.

Pixiv ID: 89228733

The magic circle looks like a complete graph with n vertices, where m edges are colored red and other edges are colored blue. Note that a complete graph is a simple undirected graph in which every pair of distinct vertices is connected by a unique edge.

Mona realizes that if she chooses four different vertices such that the six edges between these four vertices are of the same color, she will get a *key* from the magic circle. If the color is red, she will get a *red key*, and if the color is blue, she will get a *blue key*.

Base on the information written in the ancient books Mona has read, the magic power of the ancient magic circle is the absolute difference between the number of *red keys* and the number of the number of *blue keys* she can get from the magic circle.

Mona needs your help badly, since calculating the magic power of the magic circle is really a tough job.

Input

There is only one test case in each test file.

The first line of the input contains two integers n and m ($4 \le n \le 10^5$, $0 \le m \le \min(\frac{n(n-1)}{2}, 2 \times 10^5)$) indicating the number of vertices and the number of edges colored red of the ancient magic circle.

For the following m lines, the i-th line contains two integers u_i and v_i ($u_i < v_i$) indicating a red edge connecting vertices u_i and v_i. It is guaranteed that each edge appears at most once.

Output

Output one line containing one integer indicating the magic power of the ancient magic circle.

Example

standard input	standard output
7 6 1 2 1 3 1 4 2 3 2 4 3 4	3

Note

For the sample case, there is only one *red key* $(1, 2, 3, 4)$ and there are four *blue keys* $(1, 5, 6, 7)$, $(2, 5, 6, 7)$, $(3, 5, 6, 7)$ and $(4, 5, 6, 7)$ in the ancient magic circle, thus the magic power of the magic circle is $|1-4| = 3$.

Problem L. Secret of Tianqiu Valley

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

In the north tower of Tianqiu Valley's ruins, there are some flame torch puzzles and Lumine the traveler is facing the last and the hardest one.

Source: Genshin Impact Official

There are n torches in a circle and some torches have been ignited initially. The i-th and the (i mod $n+1$)-th are adjacent for all $1 \le i \le n$.

To solve the puzzle, all the torches should be ignited. In each move, Lumine can ignite an extinguished torch, and the status of the adjacent torches will be reversed affected by the supernatural. That is, each of the adjacent torches will be ignited if it is currently extinguished, or be extinguished if it is currently ignited.

Time is money, Lumine wants to solve the puzzle in $2n$ moves or determine that the puzzle is unsolvable.

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 of the input contains an integer n ($3 \le n \le 10^5$) indicating the number of torches in the circle.

The second line contains a binary string $s_1 s_2 \cdots s_n$ of length n ($s_i \in \{\text{'0'}, \text{'1'}\}$). If $s_i =$ '0' the i-th torch is extinguished initially; If $s_i =$ '1' the i-th torch is ignited initially. It is guaranteed that not all the torches have been ignited initially.

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

Output

If the puzzle is unsolvable, output "0" (without quotes).

Otherwise, output an integer k ($1 \le k \le 2n$) in the first line indicating the number of moves Lumine needs to solve the puzzle. Then output a line containing k integers $t_1, t_2, \cdots, t_k$ separated by a space,

where t_i indicating that Lumine will ignite the t_i-th torch in the i-th move. If there are multiple answers print any of them.

Example

standard input	standard output
2 5 00000 3 001	7 2 5 1 2 3 4 2 0

Note

For the first sample test case, the status of the torch will change like this: 00000 $\rightarrow$ 11100 $\rightarrow$ 01111 $\rightarrow$ 10110 $\rightarrow$ 01010 $\rightarrow$ 00100 $\rightarrow$ 00011 $\rightarrow$ 11111.

Problem M. Windblume Festival

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

The Windblume Festival in Mondstadt is coming! People are preparing windblumes for Barbatos and for those they love and adore. The Windblume Festival is also an opportunity to improve the relationships people have.

Source: Genshin Impact Official

During the festival, a famous game will be played every year, invented by Jean, the Acting Grand Master of the Knights of Favonius. In the game, n players numbered from 1 to n stand in a circle, each holding an integer with them. Each turn, one player will be removed. The game will end when there is only one player left.

For each turn, let k be the number of players remaining and a_i be the integer player i holds. Two adjacent players, x and (x mod $k+1$) are selected and player (x mod $k+1$) is removed from the game. Player x's integer will then change from a_x to $(a_x - a_{x \bmod k+1})$. Player y in this turn will become player $(y-1)$ in the next turn for all $x < y \le k$, though the integer they hold will not change.

Jean wants to know the maximum possible integer held by the last remaining player in the game by selecting the players in each round optimally.

Input

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

The first line contains one integer n ($1 \le n \le 10^6$) indicating the initial number of players.

The next line contains n integers a_i ($-10^9 \le a_i \le 10^9$) where a_i is the integer held by player i at the beginning.

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

Output

For each test case output one line containing one integer indicating the maximum possible integer.

Example

standard input	standard output
5 4 1 -3 2 -4 11 91 66 73 71 32 83 72 79 84 33 93 12 91 66 73 71 32 83 72 79 84 33 33 93 13 91 66 73 71 32 83 72 79 84 33 33 33 93 1 0	10 713 746 779 0

Note

For the first sample test case follow the strategy shown below, where the underlined integers are the integers held by the players selected in each turn.

$\{\underline{1}, -3, 2, \underline{-4}\}$ (select $x = 4$) $\rightarrow \{-3, \underline{2, -5}\}$ (select $x = 2$) $\rightarrow \{\underline{-3, 7}\}$ (select $x = 2$) $\rightarrow \{10\}$.