Problem A. Apples

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

Helena's job is collecting apples in the garden. There are n apple trees (numbered from 1 to n) and m trails (numbered from 1 to m) in the garden. Each trail connects two different apple trees. Suppose that apple tree i has d_i incident trails. Every day Helena collects apples according the following procedure every.

1. Helena randomly moves to a tree with probability of i-th tree to be selected equal to $\frac{d_i}{2m}$.
2. Helena picks a trail starting at her current position uniformly at random, then she moves to the other end of the trail.
3. Collect apples from the tree at her position. She collects a_i apples from a tree with index i on this step. Note that the apples are collected from a tree each time this tree is visited.
4. Repeat the previous two steps for k times.

Write a program to compute the expected number of apples daily collected by Helena.

Input

In first line, there is an integer T ($T \leq 20$) indicating the number of test cases.

First line of the input contains three integers n, m, k ($1 \leq n, m, k \leq 10^5$) separated by blanks in a line. There are n apple trees and m trails. k is the parameter in Helena's daily collecting procedure. In the second line of each test case, there will be n numbers a_i ($1 \leq a_i \leq 100$) indicating how many apples should be collected in the third step of the procedure. In the j-th of the following m lines, there will be two numbers u_j and v_j ($1 \leq u_j, v_j \leq n$) indicating that the j-th trail ends at tree u_j and tree v_j.

Output

Print the expected number of apples daily collected by Helena. An absolute error of 5×10^{-3} is acceptable.

Example

standard input	standard output
3 4 2 2 3 4 1 2 1 2 2 3 3 1	5.75

Problem B. Build The Road Network

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

The traveling salesman arrives in the kingdom Lazyland. He plans to start from the capital, visit every city and then return to the capital. He knows coordinates of each city, but no roads have been built yet. It is obviously no way to visit all cities if some city is not connected to all the others.

Assume there are n cities. Their coordinates are $(x_1, y_1), \cdots, (x_n, y_n)$, and (x_1, y_1) is the capital. The king is going to help the salesman. He will order to build the roads for you. However, the people in the city are too lazy.

Since the engineers are lazy, they can only plan a road to directly connect two destinations (x_i, y_i) and (x_j, y_j) using only horizontal and vertical segments. Therefore, its length is $|x_i - x_j| + |y_i - y_j|$.

Because workers are too lazy, they will not build more than $n - 1$ roads for the salesman — they heard that $n - 1$ roads are enough to travel to all destinations. Note that the salesman can only change roads at the cities.

Salesman may select roads to be built. Can you tell me how long is the shortest route for him to visit every destination and return to the capital?

Input

First line of the input contains one integer $n (n \leq 10,000)$ representing the number of destinations. The next n lines contain two integers x, y $(-1,000 \leq x, y \leq 1,000)$, representing the coordinates of the destinations. The coordinates of the capital are given in first of those lines.

Output

Print length of the shortest route for the salesman to visit every destination and return to the origin.

Example

standard input	standard output
3 1 1 2 2 3 3	8
4 2 1 -1 2 -2 -1 1 -2	24
6 1 2 2 3 2 2 3 4 4 3 3 1	16

Problem C. Coefficients

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

There is a secret polynomial $f(x)$. Its degree is less than n, and its coefficients are non-negative integers less than 1000000007 ($10^9 + 7$). That is, $f(x)$ can be written as $c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ where $c_0, \ldots, c_{n-1} \in \{0, \ldots, 10^9 + 6\}$. We will not tell you what these coefficients $c_0, \ldots, c_{n-1}$ are. Instead, we give you n point-value pairs $(x_1, f(x_1) \bmod (10^9+7), \cdots, (x_n, f(x_n) \bmod (10^9+7))$ where $x_1, \ldots, x_n$ are distinct positive integers no more than 100.

It is interesting that you can compute all coefficients from these point-value pairs. However, you are unable to determine what c_0 is if you only know $n - 1$ point-value pairs. It's time to compute the coefficients!

Input

The first line of the input contains an integer n ($n \leq 50$) indicating the number of point-value pairs. The i-th of the following n lines contains two integers x_i and r_i where $x_i \in [1, 100]$ and $r_i \equiv f(x) \bmod 10^9+7$.

Output

For each test case, output the coefficients $c_0, c_1, \ldots, c_{n-1}$ of function f. Separate them by blanks.

Examples

standard input	standard output
2 1 2 2 3	1 1
6 1 3 2 35 3 247 4 1029 5 3131 99 509900536	1 1 0 0 0 1

Problem D. Destiny

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

Odin is one of the most revered god in Norse mythology. It is known that his thirst for wisdom is unending and that he would sacrifice anything for it. In order to obtain wisdom, he hanged himself, wounded himself with a spear and even stopped eating and drinking for nine days and nine nights. Most famously, he took out one of his eyes for the right to drink in Well of Urd, also known as the Well of Destiny. The well itself is located at the root of Yggdrasil, the tree that held the nine worlds in its branches. You see, this is how destiny works on Norse Mythology. The well is the symbol of past, and all present worlds grow from it. By altering the past, one can retroactively influence the present. Being able to manipulate that is the basics of magic. In theory, Odin obtained the power to shape the cosmos as he wishes by completing these rituals.

If one wants to follow Odin's path and become as powerful as he is, it is theoretically possible. One needs only to follow his path. There are n rituals that must be completed and one must do them sequentially. And you may find suitable location for each ritual throughout the nine worlds. However, one must note that unlike a god, mortal lives are not particularly long. To travel through the nine worlds is not as easy as one might imagine. One cannot just take a flight from Midgard to Vanaheimr. The travel needs to be carefully planned in order to be done in one's lifetime. You may start at any location.

Input

The first line of the input contains three integers n ($1 \le n \le 10$), m ($1 \le m \le 300$) and e ($1 \le e \le 4000$) on the first line indicating the number of rituals that need to be performed, the number of suitable locations for them and the number of edges connecting them respectively.

On the following line are m integers $R_0, R_1, \ldots, R_{m-1}$ ($0 \le R_i < n$) in which R_i represents which ritual can be performed at the i-th (zero-based) location. Finally, there are e lines each with three integers u, v and c ($0 \le u, v < m$ and $c \le 10^4$), meaning that the number of years it take to travel from location u to v is c. Note that all these edges are **one-directional**, since traveling between the nine worlds is not as straightforward as in Midgard (which is the world humans live in).

Output

Output the minimum number of years it will take to finish every ritual sequentially from number 0 to number $n-1$ on one line. If it is impossible to finish every ritual in such order, then output -1 instead.

Examples

standard input	standard output
3 3 2 0 1 2 0 1 20 1 2 30	50
3 5 4 0 0 1 2 2 0 2 10 1 2 100 2 3 90 2 4 900	100

Problem E. Egg Hatching

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

Frigg is very fond of Pokémon GO. She wants to pick some places to go from N tourist attractions and to hatch an egg during holiday. For tourist attraction i, Frigg has to walk A_i km from her hotel and back, and there are B_i pokémons to catch. To hatch an egg in Pokémon GO, she needs to walk at least M km. Help her to find the most efficient way such that maximizes the ratio of the total number pokémons caught to the total distance walked.

Input

First line of the input contains two integers N, M ($1 \leq N \leq 1000, 1 \leq M \leq 10^5$). N is the number of tourist attractions, and M is the minimum distance to hatch an egg. In i-th of the next N lines there will be two integers A_i and B_i ($1 \leq A_i \leq 1000, 1 \leq B_i \leq 10^4$) indicating that Frigg has to walk A_i km for the i-th tourust attraction, and there are B_i pokémons to catch.

Output

For each case, output the maximum ratio of the total number of pokémons caught to the total distance walked so that the egg will hatch on the way. If there is no way to hatch an egg, then output -1. An absolute error no more than 5×10^{-4} is acceptable.

Example

standard input	standard output
3 10 5 7 5 3 5 1	1

Problem F. Freyja's Powers

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

Freyja is a goddess of love, sex, beauty, fertility and other stuff. Different from Odin, who is a Aesir god, Freyja is a member of the Vanir. These are the two opposing factions of gods that had fought in the past in which the Aesir won. Freyja and her brother Freyr are the two guest hostages taken by the Aesir and lived in Asgard for a while. As an agreement between the two factions, Freyja may take half of the dead who died valiantly in battle to her afterlife realm of Folkvang, while the other half goes to Odin's hall of Valhalla. Freyja is a passionate seeker of pleasure and thrill. However, she is also a powerful practitioner of the shamanic form of magic, seidr. Seidr magic works by discerning the weave of destiny and reweaving part of it to manipulate desires, health and prosperity of others.

Although we humans are not as knowledgeable as Freyja, but practicing seidr ourselves is not entirely impossible. We know only a few things about the patterns of destiny and can only access a fairly small part of it (otherwise we will get TLE), but there is something we can do. The weave of destiny basically looks like a lot of points on a plane and they can be connected to form edges. Because we are just mere humans, we can operate only on a 2-dimensional plane. So among these points, one of the most basic way of connecting them is to make them all into triangles. Note that two edges may not cross over each other! More formally, you will be given first a convex polygon as the boundary of the scope of your manipulation. (This is a very important step in the practice of magic, to set the boundary. Disaster often happens when one neglects this basic step.) And then you will be given a set of points. With these points, you will have to cut the polygon into several triangles by connecting points to points and forming edges. However, we don't want just any triangulation. In order to produce the best result, we need a triangulation that maximizes the smallest angle in all those triangles.

Input

In first line of the input there are two numbers n and m. n represents the number of points which made up of the boundary, and m is the number of points inside it. You may assume $n + m \leq 50$ Note that the points inside the boundary may lie exactly on the boundary. The next line will contain $2n$ integers $x_1, y_1, x_2, y_2, \ldots, x_{n-1}, y_{n-1}$ separated by blanks, representing the boundary $(x_1, y_1), \ldots, (x_n, y_n)$ in counter-clockwise order. Finally there will be $2m$ integers on the last line in the same format, representing the points inside the boundary or right on the boundary. All coordinates are within the range -100 to 100.

Output

Output the degree of the smallest angle on one line. An absolute error less than 5×10^{-3} is acceptable.

Example

standard input	standard output
3 0 0 0 1 0 0 1	45.00
4 2 0 0 2 0 2 2 0 2 1 1 0 1	45.00

Problem G. Grid and Triangles

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 second
Memory limit:	256 mebibytes

A triangle is a shape which can be formed by connecting three non-colinear points with straight lines.

You are Given a rectangular grid formed by n distinct horizontal lines and m distinct vertical lines. How many different triangles can be formed using the points on the intersections?

Two triangles are considered different if sets of their vertices differ.

Input

In first line of input, there is an integer T ($1 \leq T \leq 300$) indicating the number of test cases. Each of the following n lines contains a test case. Each test case has two positive integers n and m separated by a space. Both of them are no more than 100, and there product nm is no more than 1024.

Output

For each test case, print the number of triangles.

Example

standard input	standard output
2 2 2 3 3	4 76

Problem H. Happy Little Tree

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

Kappa has a happy little rooted tree with n nodes. Kappa's friend Keepo has a favourite node on this tree. Kappa doesn't mind letting Keepo play with his tree, but he has his own rule: he only allows Keepo to traverse his tree by pre-order depth-first search. In addition, nodes are indexed by 0 to $n-1$, and each node is colored either black or white. These also limit the order of traversal. Here's the pseudocode of the traverse rule:

```
DFS(tree T):
    visit T's root u
    if(u is white):
        for(all u's child v, in ascending order by their index):
            DFS(subtree rooted at v)
    else:
        for(all u's child v, in descending order by their index):
            DFS(subtree rooted at v)
```

The root has index 0, and each non-root node has greater index than its parent. Keepo like node $n-1$ the most, but he doesn't want to spend too much time on traversing the tree. Therefore he asks you to tell him how many nodes will he visit before he reaches his favourite one. What make things even more complicated is that Kappa and Keepo change their mind frequently. Sometimes Kappa choose to change the colors of all nodes in a subtree, sometimes Keepo change his favourite node. After each change, you should also tell Keepo the new answer.

Input

The first line of the input contains an integer n ($1 \le n \le 10^5$), the number of nodes. At the beginning, all nodes are white. The second line contains $n-1$ integers $p_1, p_2, \ldots, p_{n-1}$. p_i ($0 \le p_i < i$) is the index of node i's parent. The third line contains an integer q ($1 \le q \le 10^5$), the nubmer of changes. There are q lines following. In each line, there contains a character and a number u ($0 \le u < n$). If the character is '`a`', it means Kappa changes the colors of all nodes in the subtree rooted at node u. If the character is '`e`', it means Keepo changes his favourite node to node u.

Output

For each test cases, print $q+1$ lines. The first line contains the nubmer of node visited before node $n-1$. The next q lines contains the new answer after each change.

Example

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

Problem I. Integration of Patterns

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

Define a *pattern* as a sequence of lowercase English letters, and characters '`*`' and '`?`'.

Let us say that a string s conforms to a pattern a if it is possible to replace '`*`' characters in a with several (possibly none) lowercase English letters, and characters '`?`' with a lowercase English letter each so that to obtain the string s (for example, string "`barcelona`" conforms to the pattern "`b*a?ce?*a`").

We will say that a pattern c is an *integration* of two patterns a and b if the following conditions hold:

1. If a string s conforms to a or b, then s must conformsto c.
2. c must contain minimal possible number of characters '`*`'.
3. Among all patterns satisfying the above conditions c must be one of the longest.
4. Among all patterns satisfying the above conditions c must contain minimal possible number of '`?`'

Given patterns a and b, print any pattern that is their integration.

Input

First line of the input contains pattern a, $1 \le |a| \le 1000$. Second line contains pattern b, $1 \le |b| \le 1000$. It is guaranteed that both given strings are correct patterns.

Output

Print one string — integration of given patterns. If there is still more than one solution satisfying conitions 1) - 4), print any of them.

Examples

standard input	standard output
barcelona bootcamps	b????????
coders code	code*
icpc* in?	i??*

Problem J. Jewel Game

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

Alice and Bob are playing a game with some unbreakable jewels. Initially there are some piles of jewels, and they'll take turns splitting a pile. In each turn, a player should choose a pile with more than one jewel and split it into at least two piles of equal size. Alice moves first, and the player who can't make a valid move loses. Who will win the game if both of them play optimally?

Input

Input contains two lines. The first line contains an integer n ($n \leq 10000$), which is the number of piles. The second line contains n integers $a_1, a_2, \ldots, a_n$ ($a_i \leq 10^9$), giving the number of jewels in each pile.

Output

For each test case, print "`Alice`" if Alice will win or "`Bob`" otherwise in one line.

Example

standard input	standard output
2 3 15	Alice
3 3 5 15	Alice
4 3 5 15 21	Bob

Problem K. Kaprekar's constant

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

Consider the following operation on a 4-digit number (leading zeros are allowed):

1. Rearrange the digits in descending order to get a 4-digit number X.
2. Rearrange the digits in ascending order to get a 4-digit number Y.
3. The result is $X - Y$.

The Indian mathematician Kaprekar found that if we apply the operation repeatedly on any 4-digit number using at least 2 different digits, it will reach the fixed point 6174 in at most 7 iterations. The fixed point 6174 is known as Kaprekar's constant. Take 1234 for example. After an iteration, it becomes $X - Y = 4321 - 1234 = 3087$. After the second iteration, it becomes $8730 - 0378 = 8352$. After the third iteration, it becomes $8532 - 2358 = 6174$. 6174 is a fixed point because $7641 - 1467 = 6174$. Note that if we choose a number whose digits are the same, the result will be 0000. For example, $3333 - 3333 = 0000$.

How about the 5-digit case? Actually some numbers don't stop at a fixed point. For instance, 87433 falls into $\{53955, 59994\}$, a cycle of length 2. Given a 5-digit number, can you figure out what will the result be after 999999 iterations?

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), the number of test cases. There are T lines following. Each line contains a 5-digit integer.

Output

For each testcase, output the answer in a line. Leading zeros must be printed.

Example

standard input	standard output
3 87433 53955 33333	53955 59994 00000

Problem L. Little Triangle

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

There are so many methods to calculate the area of a triangle. For example, the area of a triangle of base b and height h is $\frac{bh}{2}$, and it can be also represented as $\sqrt{s(s-a)(s-b)(s-c)}$ where a, b, c are the lengths of its edges and $s = \frac{a+b+c}{2}$. There are also a lot of problems related to the areas of triangles. You probably have solved most of them but not this one.

This time, you have to find the smallest triangle in terms of area among a set S of infinitely many triangles. Sounds hard? Just let me make the problem a little bit simpler by adding some limitation on S. The triangles in S are exactly all the triangles which have vertices in a set L of infinite points in 2D plane. Still too hard? Let me make it even simpler. The set L is determined by n vectors $v_1 = (x_1, y_1), \ldots, v_n = (x_n, y_n)$ where $x_1, \ldots, x_n, y_1, \ldots, y_n$ are integers. L is defined as follows.

- $(0, 0) \in L$.
- If $(a, b) \in L$, then $(a + x_i, b + y_i) \in L$ and $(a - x_i, b - y_i)$ for every $i \in \{1, \ldots, n\}$.

That is, any point $(x, y) \in L$ if and only if (x, y) can be written as an integer linear combination of $v_1, \ldots, v_n$. More precisely, $(x, y) = (z_1x_1 + \cdots, z_nx_n, z_1y_1 + \cdots + z_ny_n)$ for some integers $z_1, \ldots, z_n$.

Now, I believe the problem is easy enough to you. Let's get it done.

Input

In first line of input, there is an integer T ($T \leq 500$) indicating the number of test cases. The first line of each test case contains an integer n ($n \leq 100$) indicating the number of vectors. The i-th of the following n lines contains two integers x_i and y_i where $x_i, y_i \in [-10^9, 10^9]$ and $v_i = (x_i, y_i)$. All points in the test case can be represented by an integer combination of $v_1, \ldots, v_n$. I.e., it can be written as $(z_1x_1 + \cdots + z_nx_n, z_1y_1 + \cdots + z_ny_n)$ for some integers $z_1, \ldots, z_n$.

Output

For each test case, output the area of the smallest triangle with absolute error 0.1 or less. If these points cannot form any triangle, output -1 instead.

Example

standard input	standard output
2 2 1 0 2 0 3 1 3 3 1 4 4	-1 4.0