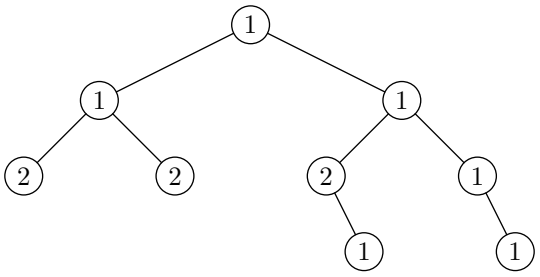
Problem A. Almost Balanced Tree

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

Consider a binary tree, each node has a weight equal to 1 or 2. The weight of a subtree is the sum of weights of all nodes in the subtree. The weight of an empty tree is 0.

The binary tree is *almost balanced* if for each node, the difference of weights of its children subtrees is at most 1 (if one of the children is missing, its weight is considered to be 0).

Here is an example of an almost balanced binary tree:



Your task is to build any almost balanced binary tree with exactly A nodes of weight 1 and B nodes of weight 2, or to say that it is impossible.

Input

The input contains two non-negative integers A and B ($1 \le A + B \le 100\,000$).

Output

Assign indices from 1 to $A + B$ to the nodes of your tree, node 1 should be the root of the tree. Output $A + B$ lines, one for each node. Each line should contain three integers — the weight of the node, and the indices of the left and the right children of the node. If the corresponding child is missing, output 0 instead.

If it is impossible to construct an almost balanced tree, output -1.

If there are multiple possible answers, output any one of them.

Examples

standard input	standard output
6 3	1 2 5 1 3 4 2 0 0 2 0 0 1 6 7 2 0 8 1 0 9 1 0 0 1 0 0
0 2	-1

Problem B. Brain-teaser

Input file:	standard input
Output file:	standard output
Time limit:	5 seconds
Memory limit:	512 megabytes

There is a type of a brain-teaser known as *cryptoarithmetics*. The classic example, published in the July 1924 issue of Strand Magazine by Henry Dudeney, is:

```
    S E N D
+   M O R E
-----------
= M O N E Y
```

A solution to such a puzzle is an assignment of letters to decimal digits from 0 to 9 that satisfies the following constraints:

- Each letter must represent a different decimal digit.
- The leading digit of each number must not be zero.
- The result must be a correct equality — the sum of the first two decimal numbers is equal to the third one.

The only solution to the above puzzle is O = 0, M = 1, Y = 2, E = 5, N = 6, D = 7, R = 8, and S = 9, producing the following equality: $9567 + 1085 = 10652$.

A *good* brain-teaser, like the classic one, has a unique solution. You are given the first two words in a brain-teaser and your task is to find all the possible third words from the given dictionary that produce a good brain-teaser with a unique solution.

Input

The first and the second lines of the input contain a word each — two addends of the brain-teaser. The third line of the input contains an integer n — the number of words in the dictionary, followed by n lines with the dictionary words. The words in the dictionary are lexicographically ordered.

All words in the input consist of 2 to 15 uppercase letters. All the tests, with the exception of the first test, use the same dictionary of 279 496 Collins Dictionary Scrabble Words (2019). The first two words come from this dictionary, too. Note that you can find the second test with the full dictionary inside the downloadable file with the problem examples. It is provided together with the problem statements.

Output

On the first line output a single integer m — the number of words from the dictionary that produce a good brain-teaser with the first two given words. On the next m lines output the words in the same order as in the dictionary.

Examples

standard input	standard output
SEND MORE 3 FUN HONEY MONEY	1 MONEY
DUB UPSPEAK 279496 ... not shown ...	2 UPMAKER UPTAKEN

Problem C. Color the Tree

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

Christina has a rooted tree with n vertices. Initially, all vertices are colored green, except for the root, which is colored red. Christina thinks that the tree is *beautiful* if two rules are satisfied:

- The root is colored red.
- If the vertex is colored red, all vertices on the shortest path between it and the root are also red.

Christina repeatedly performs the following operation on the tree — chooses a vertex and changes its color (if it was red, colors it green; if it was green, colors it red). The following rules must be satisfied while performing the operations:

- The tree should stay beautiful.
- The coloring of vertices should be unique. That means there is no moment in the past when each vertex had the same color as it has right now.

Your task is to help Christina build the longest possible sequence of operations following the rules.

Input

The first line contains an integer n ($1 \le n \le 20$) — the number of vertices in the tree.

The second line contains $n - 1$ integers p_i ($1 \le p_i \le i$ for $1 \le i \le n - 1$), denoting parent vertices in the tree. The vertices in the tree are numbered from 1 to n, the root has number 1, the i-th vertex has parent p_{i-1} for $2 \le i \le n$.

Output

On the first line output an integer m — the maximum number of operations.

On the second line output m integers o_i ($2 \le o_i \le n$). o_i is the number of the vertex that changes color during the corresponding operation.

If there are several possible longest sequences, output any one of them.

Examples

standard input	standard output
4 1 1 1	7 4 3 4 2 4 3 4
4 1 2 3	3 2 3 4
6 1 1 2 2 2	17 3 2 3 6 3 5 3 6 3 4 3 6 3 5 3 6 3
5 1 2 1 1	11 5 4 5 2 5 4 5 3 5 4 5
5 1 1 2 3	8 3 5 2 5 3 4 3 5

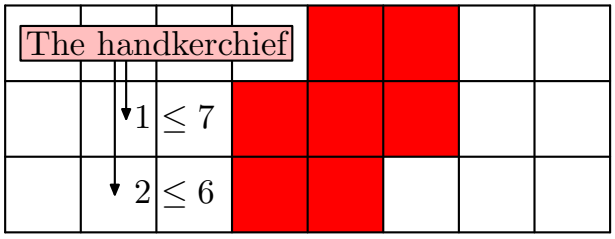
Problem D. Down We Dig

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 megabytes

Dina and Dima are young archeologists exploring an ancient mosaic-covered staircase attributed to Dacian culture (presumably, king Decebalus himself) in Danube Delta (modern Dobruja region).

Each step is covered with 8 pieces of mosaic, each being either white or red. Each morning they dig out exactly one step of the stairway, obviously from top to bottom.

Each afternoon while walking down the staircase towards the working area after lunch, they play a game. They put (very carefully) a handkerchief on the topmost step. Then they make moves in turn, starting with Dina. Each move, a player moves the handkerchief down a few steps. It is only allowed to move the handkerchief from one step to a lower step if the distance between these steps is less than or equal to the number of their common mosaic pieces (pairs of the same color in the same positions). The player who can not make a move loses today's game.



For example, here Dina can move the handkerchief from the topmost step to the middle one (because $1 \le 7$) or to the bottom one (because $2 \le 6$).

For each afternoon, find out who wins the game if they both play optimally.

Input

The first line contains an integer n ($1 \le n \le 300\,000$) — the height of the staircase.

Each of the next n lines contains 8 characters 'W' or 'R' — the descriptions of steps from top to bottom.

Output

Output a line with n digits, one digit for each afternoon game. 1 means that Dina wins, 2 means that Dima wins.

Examples

standard input	standard output
3 WWWWWWWW RRRRRRRR WWWWWWWW	221
7 WWWWRRWW WWWRRRWW WWWRRWWW WWRRRWWW WWRRWWWW WRRRWWWW WRRWWWWW	2111121

Problem E. Easy Measurements

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

Edward was tasked to measure the pumping rate of two water pumps. To do so, he used the pumps to pump water into a water tank and checked how much water was pumped into the tank in a specific time.

He found out that the first pump pumps a liters of water in b seconds, and the second pump pumps c liters of water in d seconds. He also discovered that when both pumps are used at the same time, together they pump b liters of water in d seconds.

Unfortunately, Edward has spilled some water on his records, so now he can't recover the values a and c. However, he remembers that these values were positive integers. Now he wonders how many ways are there to choose the values a and c that are consistent with his measurements.

Input

The first line contains an integer n ($1 \le n \le 1000$) — the number of test cases.

Each of the next n lines contains two integers b and d ($1 \le b, d \le 10^9$).

Output

For each test case, output a single integer — the number of ways to choose a and c. Output each answer on a separate line.

Example

standard input	standard output
3 9 6 40 60 60 40	4 13 29

Note

In the first test case, the possible values are $a = 3$, $c = 7$; $a = 6$, $c = 5$; $a = 9$, $c = 3$; and $a = 12$, $c = 1$.

Problem F. Find a Square

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	512 megabytes

Frank likes square numbers. That is numbers, which are the product of some integer with itself. Also Frank likes quadratic polynomials. He even has his favorite one: $p(x) = a \cdot x^2 + b \cdot x + c$.

This morning Frank evaluated his favorite quadratic polynomial for n consecutive integer arguments starting from 0 and multiplied all the numbers he got.

If the resulting product is a square, his day is just perfect, but that might be not the case. So he asks you to find the largest square number which is a divisor of the resulting product.

Input

The only line of the input contains 4 integers a, b, c, n ($1 \le a, b, c, n \le 600\,000$).

Output

Find the largest square divisor of $\prod_{i=0}^{n-1} p(i)$. As this number could be very large, output a single integer — its remainder modulo $10^9 + 7$.

Examples

standard input	standard output
1 1 1 10	74529
1 2 1 10	189347824

Note

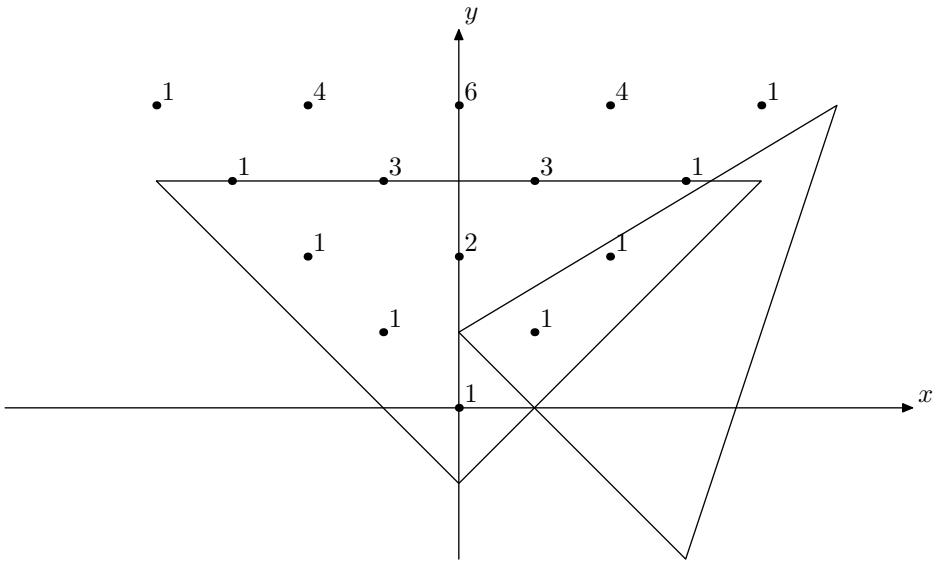
In the first example, the product is equal to $1 \cdot 3 \cdot 7 \cdot 13 \cdot 21 \cdot 31 \cdot 43 \cdot 57 \cdot 73 \cdot 91 = 2893684641939 = 38826291 \cdot 273^2$.

Problem G. Geometrical Combinatorics

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

Grace is developing a brand new theory of geometrical combinatorics — a study about geometrical properties of combinatoric objects.

Consider two triangles on plane — a Pascal's triangle and an ordinary triangle. Pascal's triangle is drawn with it's root at point (0, 0), and two sides along diagonals of upper-halfplane quarters. Formally, there are 1's written in points (i, i) and $(-i, i)$, and between them at point $(-i + 2k, i)$ there is a number equal to the sum of numbers at $(-i+2k+1, i-1)$ and at $(-i+2k-1, i-1)$ for all k from 1 to $i-1$. An ordinary triangle is drawn as just a triangle with vertices at (x_A, y_A), (x_B, y_B), (x_C, y_C).



Grace defines an *intersection value* of Pascal's triangle and an ordinary triangle as the sum of values of Pascal's triangle inside or on the border of the ordinary triangle. Can you develop a program that calculates this intersection value?

Input

On the first line there is an integer t ($1 \le t \le 5$) — the number of tests to process. Each of the next t lines contains 6 integers x_A, y_A, x_B, y_B, x_C, y_C ($-10^6 \le x_A, y_A, x_B, y_B, x_C, y_C \le 10^6$). Three points in each test do not lie on a line.

Output

For each test output an integer — the intersection value modulo $10^9 + 7$.

Example

standard input	standard output
2 0 -1 -4 3 4 3 5 4 0 1 3 -2	15 2

Problem H. Hit the Hay

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 megabytes

Some consider putting a baby to sleep to be an art, but this problem will show it is all just maths.

Consider a night during which a parent is trying to put a baby to sleep. An alarm clock will sound at the end of the night, and the parent is not allowed to change the alarm time, so the length of the night is fixed at k hours from now.

The baby can be in one of the three states: state 0 means the baby is awake, state 1 means the baby is in light sleep, and state 2 means the baby is in deep sleep. The baby starts in state 0, and the change in a state happens continuously rather than discretely. You are given three numbers p_0, p_1, and p_2. Whenever the baby is in state i, the probability that no state change will happen in the next x hours is p_i^x, where x is a positive real number. In other words, the time before the next state change is picked from the *exponential distribution* with the cumulative distribution function of $1 - p_i^x$.

Whenever a state change does happen, if the baby was in state 0, it will always switch to state 1; if the baby was in state 2, it will also always switch to state 1; if the baby was in state 1, it will switch to state 0 with the probability q_0 and to state 2 with the probability $1 - q_0$.

The parent decides when to go to sleep themselves. However, if the baby is in state 0, it will cry and wake the parent up, so the parent can only be asleep if the baby is in state 1 or 2. The parent can choose to still stay awake even if the baby is in one of those states. If they do stay awake, they can:

- see which of the three states the baby is in;
- prevent the baby from waking up: if the baby decides to switch from state 1 to state 0 according to the above rules, and the parent is not asleep, then the baby will be comforted and will stay in state 1 instead.

The parent can decide to go to sleep arbitrarily, for example using the current state of the baby or the current time to make this decision. However, if they do go to sleep, then they will be asleep until either the baby wakes up (goes to state 0), or the alarm clock sounds at the end of the k hours. If they get woken up by the baby waking up, then they can later decide to go to sleep again arbitrarily.

What is the maximum expected number of hours of sleep the parent can get if they decide to go to sleep in the optimal fashion?

Input

The first line of the input contains an integer t ($1 \le t \le 1000$) — the number of test cases.

The next t lines describe test cases, each contains five floating-point numbers with exactly *one* digit after the decimal point, in the following order: k, p_0, p_1, p_2, q_0 ($0.1 \le k \le 10$; $0.1 \le p_0, p_1, p_2, q_0 \le 0.9$).

Output

Output t lines with a floating-point number on each line — the maximum expected amount of sleep for each test case. Your outputs will be considered correct if they are within 10^{-9} absolute difference from the answers.

Example

standard input	standard output
2 10.0 0.5 0.5 0.5 0.5 8.0 0.1 0.9 0.9 0.1	6.5990202123649855 7.540407031059442

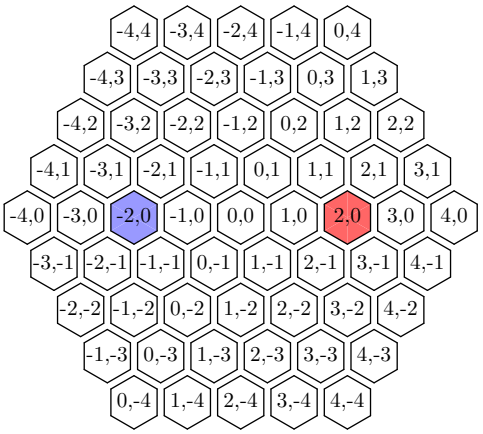
Problem I. Interactive Knockout

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

This is an interactive problem. You are to defeat a randomly moving jury in a series of independent rounds of the game.

The game is played on a hexagonal field with axial coordinates. The field is bounded by a hexagon with vertices in cells $(n, 0)$, $(0, n)$, $(-n, n)$, $(-n, 0)$, $(0, -n)$, and $(n, -n)$. In all test cases, except for the sample test case in this statement, which is not present in the real test set, $n = 20$.

There are two players — you and the jury. You start in the cell $(-n/2, 0)$ and the jury starts in the cell $(n/2, 0)$. Players take turns, you move first.



A field for $n = 4$ (the blue cell is your starting cell and the red cell is the starting cell for the jury).

At each turn, the player moves to any cell adjacent by side, which does not contain the opponent and has not been destroyed. After that, the previously occupied cell is destroyed and is not available to players on the following turns anymore. The player who cannot move to any adjacent cell loses the game.

The jury did not come up with any smart algorithm to play this game, so they've decided to move equiprobably randomly to any valid adjacent cell on each turn.

You need to show your complete dominance — win all of t independent rounds of the game.

Interaction Protocol

In the first line, you are given integers t and n — the number of independent rounds of the game you need to win, and the field size ($1 \le t \le 50$; $n = 20$ except for the sample test case, which is not present in the real test set).

Each turn, you need to output a line with the direction of your move — two integers dx, dy, where $(dx, dy) \in \{(1, 0), (0, 1), (-1, 1), (-1, 0), (0, -1), (1, -1)\}$. After that, read one line in the following format:

- token "`move`" and two integers dx, dy — direction (dx, dy) where the jury moved. It is guaranteed to be chosen equiprobably randomly;
- token "`win`", if the jury doesn't have any valid cells to move to. In this case, you should immediately start playing the next game, or finish your program with the exit code 0 if all t rounds were played.
- token "`lose`", if you made an invalid move. In this case, you should finish your program with the exit code 0 to get an adequate verdict of "Wrong answer".

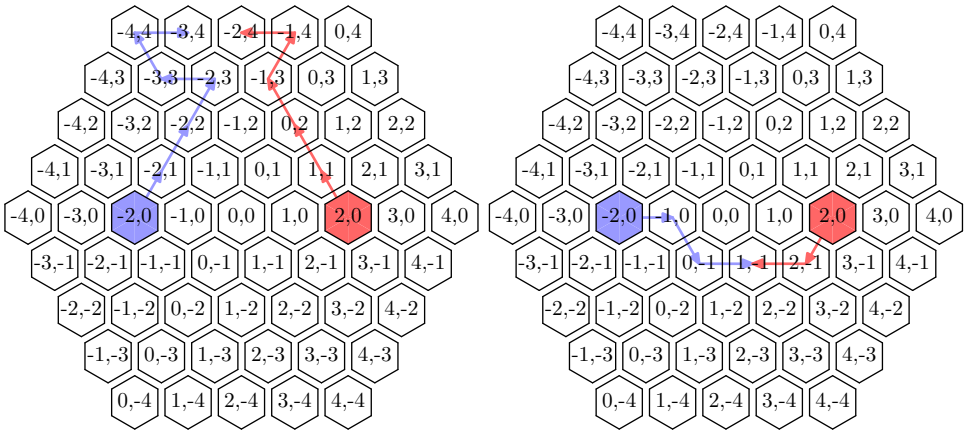
It is guaranteed that there are at most 100 tests (with at most 50 game rounds in each test) for a total of at most 5000 game rounds that you shall win. The jury uses a fixed random seed for each test.

Example

standard input	standard output
2 4 move -1 1 move -1 1 move -1 1 move 0 1 move -1 0 win move 0 -1 move -1 0 lose	 0 1 0 1 0 1 -1 0 -1 1 1 0 1 0 1 -1 1 0

Note

Note that the interaction in the sample test case results in the "Wrong answer" verdict, as only 1 round out of 2 is won. The two rounds played are shown below.



The starting player wins (on the left) and loses by making an invalid move (on the right).

Problem J. Jumping Cat

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

A city skyline is specified by n integers $d_1, d_2, \ldots, d_n$ ($0 < d_1 < d_2 < \ldots < d_n$) and n integers $h_1, h_2, \ldots, h_n$.

A skyline surface consists of n horizontal line segments, the i-th segment connects points (d_{i-1}, h_i) and (d_i, h_i), where $d_0 = 0$. Each segment is a roof of a building.

A cat (which is so small that can be considered a point) wants to get from the leftmost point of the skyline, $(0, h_1)$, to the rightmost point of the skyline, (d_n, h_n). To achieve that, the cat performs a sequence of moves. Each move is one of two types:

1. *Walk* from point (x_1, y_1) to point (x_2, y_2). Both points must belong to the same surface segment, i. e. there exists i such that $y_1 = y_2 = h_i$ and $d_{i-1} \le x_1, x_2 \le d_i$. A trajectory of a walk is a straight line segment.

2. *Jump* from point (x_1, y_1) to point (x_2, y_2). Points (x_1, y_1) and (x_2, y_2) must belong to different surface segments. A trajectory of a jump is a straight line segment and must satisfy the following constraints:
   - the distance between (x_1, y_1) and (x_2, y_2) is at most L;
   - the line segment between (x_1, y_1) and (x_2, y_2) does not intersect any of the buildings, i. e. there is no point (x, y) belonging to the segment and integer i such that $d_{i-1} < x < d_i$ and $y < h_i$.

The length of the cat's trajectory is the sum of lengths of all the moves in it. Find the shortest trajectory for the cat to get from $(0, h_1)$ to (d_n, h_n), or determine that the goal is unreachable.

Input

The first line contains two integers n and L ($1 \le n \le 50$; $1 \le L \le 100$).

The second line contains n integers $d_1, d_2, \ldots, d_n$ ($0 < d_1 < d_2 < \ldots < d_n \le 1000$).

The third line contains n integers $h_1, h_2, \ldots, h_n$ ($1 \le h_i \le 100$; $h_i \ne h_{i+1}$).

Output

Output a single floating-point number — the length of the shortest trajectory from point $(0, h_1)$ to point (d_n, h_n), or -1 if no valid trajectory exists.

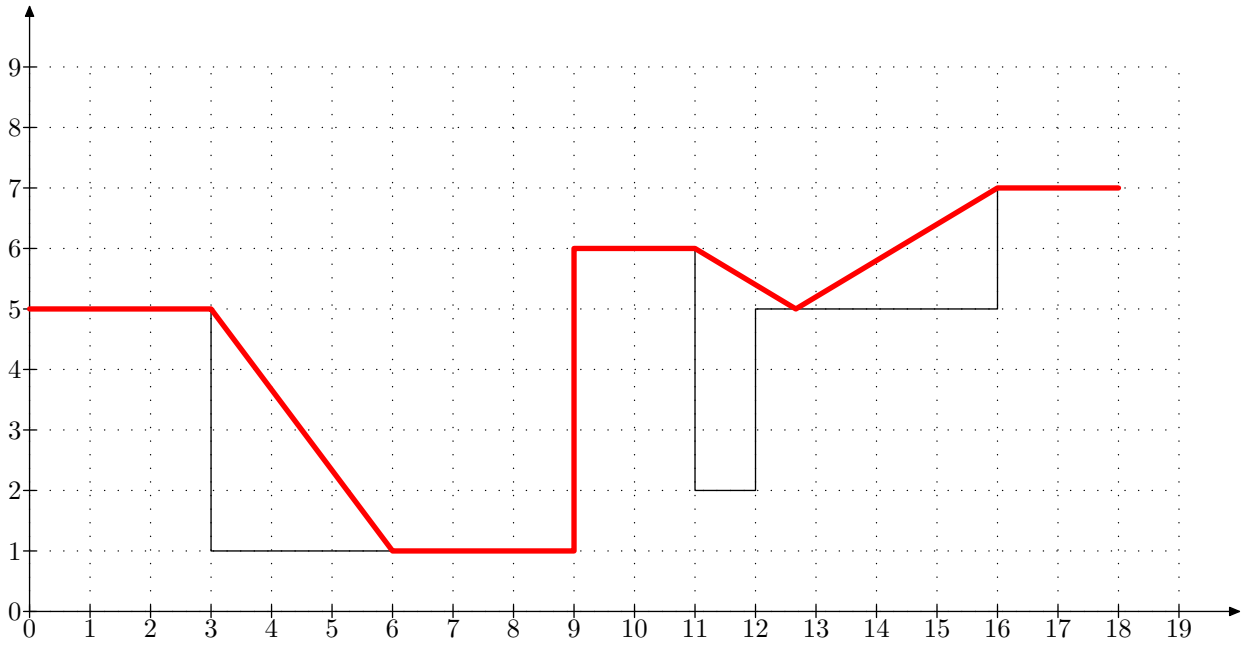
Your answer will be considered correct if its absolute or relative error doesn't exceed 10^{-9}.

Examples

standard input	standard output
6 5 3 9 11 12 16 18 5 1 6 2 5 7	25.83095189484530047
6 4 3 9 11 12 16 18 5 1 6 2 5 7	-1

Note

The picture for the first sample is shown below.



Problem K. Kate's 2021 Celebration

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

Kate did not have a good year 2020 and she is glad that it is coming to an end. She is planning to celebrate New Year 2021 on a grand scale.

Kate has decided to buy four balloons with digits `2 0 2 1` painted on them for her 2021 celebration. She went to a store's web site and has found that balloons with digits are sold in packs containing different assortments of balloons of various sizes, colors, and with different digits painted on them. Kate does not care about their size, color, or other attributes. She only cares about digits written on those balloons. Fortunately, the store has all the information about different packs of balloons that are available. Kate has managed to retrieve it through the store's REST API and extracted just the information she needs — the price and the digits that are written on balloons in each pack.

Please, help Kate with the final task of figuring out what is the cheapest pack of balloons she can buy that would get her the four digits she needs for her 2021 celebration.

Input

The first line of the input file contains an integer n ($1 \le n \le 1000$) — the number of packs of balloons available in the store.

The next n lines contain descriptions of packs, one line per pack. Each pack is described by an integer p ($1 \le p \le 10^5$) — the price of the pack in roubles, followed by a string of at least one and at most 100 digits (each digit is from `0` to `9`) — the digits on the balloons in the pack.

Output

Output a single integer — the number of the cheapest pack that Kate can buy to get the digits for her 2021 celebration. Packs of balloons are numbered starting from 1 in the order they are given in the input. If there are multiple packs with the same price, output any one of them.

Output 0 if there is no pack in the store that Kate can buy for her 2021 celebration.

Examples

standard input	standard output
4 100 9876543210 200 00112233445566778899 160 012345678924568 150 000000123456789	3
5 100 0123456789 120 0022446688 200 00224466883456789 10 0 10 1	0

Note

In the first example, 2nd and 3rd packs of balloons contain digits `2 0 2 1` and the 3rd one is the cheapest.

Problem L. Lookup Performance

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

A *binary search tree* is a rooted binary tree whose nodes store keys so that each node's key is greater than all the keys in the node's left subtree and less than those in its right subtree.

A binary search tree can be used to maintain sorted sets and allows to perform different types of queries on the set. A query that we are considering in this problem is finding the number of values in the range $[L, R]$ within the set.

Let S be the set of all keys in the binary search tree. You are given two values L and R. The query is to find the number of such $x \in S$ so that $L \le x \le R$. The following recursive function computes this value when called with the parameters `lookup(root, L, R)`, where `root` is the root of the binary search tree.

```
function lookup(v, L, R):
  if v == null:
    return 0
  if L <= v.min and v.max <= R:
    return v.count
  if v.max < L or R < v.min:
    return 0
  res = 0
  if L <= v.key and v.key <= R:
    res += 1
  res += lookup(v.left, L, R)
  res += lookup(v.right, L, R)
  return res
```

Values `v.left`, `v.right`, `v.min`, `v.max`, `v.key`, and `v.count` are the fields of the nodes of the binary search tree.

- `v.left` and `v.right` are the left and the right children of node v, respectively.
- `v.min` and `v.max` are the minimum and the maximum keys in the subtree rooted at node v.
- `v.key` is the key of node v.
- `v.count` is the number of nodes in the subtree rooted at node v.

You are given a binary search tree with integer keys. You are also given queries, each query consisting of two integers L and R. Find the number of calls of the `lookup` function that are made when `lookup(root, L, R)` is called, including the initial `lookup(root, L, R)` call itself.

Input

The first line contains an integer n ($1 \le n \le 200\,000$) — the number of nodes in the binary search tree.

The next n lines describe the nodes of the tree. The i-th of these lines contains three integers l_i, r_i, and k_i denoting the left child, the right child and the key of the i-th node. If the node does not have the left or/and the right child, the corresponding value is 0 ($l_i = 0$ or $i < l_i \le n$; $r_i = 0$ or $i < r_i \le n$; $-10^9 \le k_i \le 10^9$). The root of the tree is at node 1 and it is guaranteed to be a well-formed binary search tree.

Note that the values `v.min`, `v.max` and `v.count` are not given in the input explicitly, since they can be deduced from l_i, r_i and k_i.

The next line contains an integer q ($1 \le q \le 200\,000$) — the number of the queries.

Each of the next q lines contains two integers L and R ($-10^9 \le L \le R \le 10^9$) — the parameters given to the `lookup` function.

Output

Output q lines, the i-th line containing a single integer — the number of calls of the `lookup` function that are made for the i-th query.

Example

standard input	standard output
7 2 3 4 4 0 2 5 6 8 0 0 1 0 7 5 0 0 9 0 0 6 5 2 7 0 10 2 8 4 4 3 3	7 1 7 3 3

Problem M. Miser

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

In some non-classical University, there is going to be an opening ceremony of the cafeteria in n days. In front of the closed cafeteria, there is a sign with a number — how many days are left before the opening.

For each day out of these n, the director of the cafeteria knows all the people who are coming to the University and are going to see the sign. The director has to choose a sign with a number each day, such that each person who is coming to the University sees that the number on the sign is decreasing. The director is a typical *miser* who spends as little money as possible and wants to order the minimum possible number of different signs. Your task is to help the director find this number.

Consider the first test case: person 1 comes on days 1, 2 and 5, and person 2 comes on days 2, 3 and 4. The director can order just four signs with numbers 1, 2, 3 and 4, to put a sign with 1 on days 5 and 4, a sign with 2 on day 3, a sign with 3 on day 2, and a sign with 4 on day 1. Thus, person 1 will see the signs 4, 3, and 1 and person 2 will see the signs 3, 2, and 1.

Input

The first line of the input contains an integer n — the total number of days before the opening of the cafeteria. The next n lines contain the description of each day. The description starts with the positive integer k — the number of people that come to the University this day. This integer is followed by k distinct integers — the identifiers of the people that come.

The sum of all k over all days does not exceed 10^5. Identifiers of people are positive and do not exceed 10^5.

Output

Output one integer — the minimum possible number of different signs that have to be ordered.

Examples

standard input	standard output
5 1 1 2 1 2 1 2 1 2 1 1	4
5 1 1 1 1 1 1 1 1 1 1	5

Problem N. New Flat

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

Flato lives in Flatburg, the capital of Flatland, and he just bought a new flat! His only problem is that while the movers brought in his favorite couch, they have put it in the wrong place. Now Flato wants to fix this by moving his couch, but it is pretty long. Can you help Flato?

As with all flats in Flatland, Flato's flat is a convex polygon. His favorite couch is infinitely thin, so we would represent it as a segment. Formally speaking, we have a polygon P representing the flat and a segment AB inside the polygon representing the couch. We say that couch can *reach* position CD if there are 2 continuous functions f and g from $[0, 1]$ to the inside or the boundary of P such that $f(0) = A, f(1) = C, g(0) = B, g(1) = D$ and $|f(x)g(x)| = |AB|$ for $0 \le x \le 1$. Your task is to find the maximal possible value of the angle between the lines AB and CD for all the reachable positions CD. The angle between lines is defined as the minimum of two angles at the point of intersection, or 0 if lines are parallel.

Input

The first line of the input contains five integers n, x_A, y_A, x_B, and y_B ($3 \le n \le 50$; $-15\,000 \le x_A, y_A, x_B, y_B \le 15\,000$) — the number of vertices in P and the coordinates of the ends of the couch.

The next n lines contain two integers x and y each ($-15\,000 \le x, y \le 15\,000$) — the coordinates of the polygon vertices in counter-clockwise order.

It is guaranteed that both A and B are either inside or on the boundary of P and that the polygon is convex.

Output

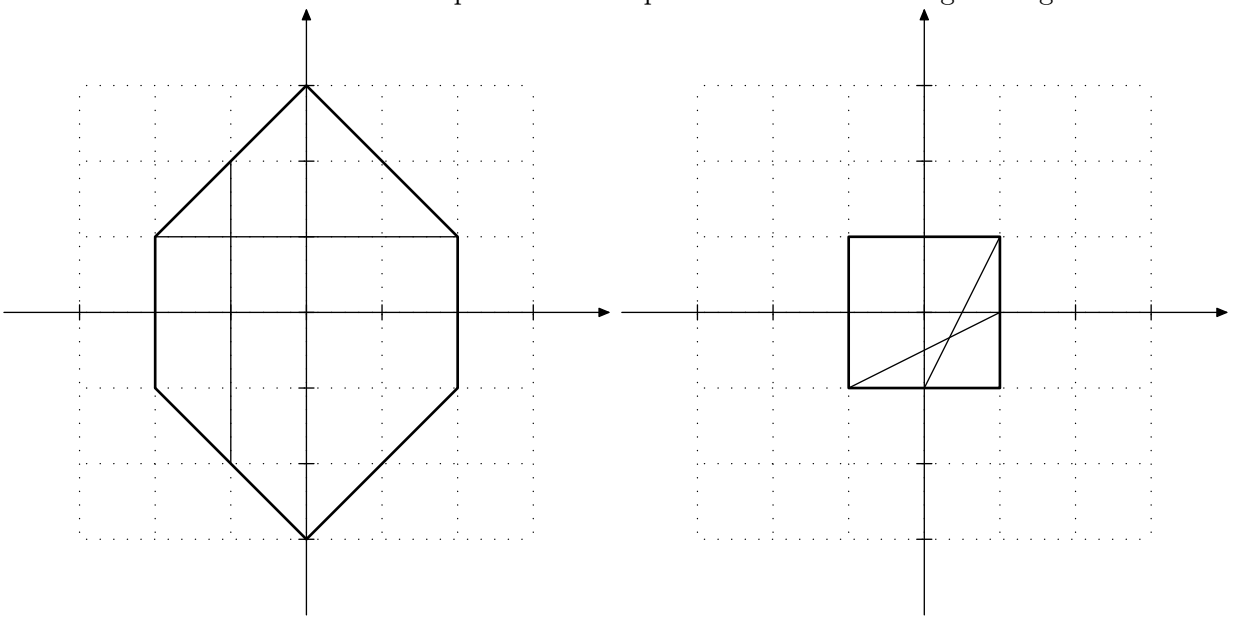
Output the maximal angle in degrees as described in the problem statement. Your output will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Examples

standard input	standard output
6 2 1 -2 1 2 1 0 3 -2 1 -2 -1 0 -3 2 -1	90
4 -1 -1 1 0 1 1 -1 1 -1 -1 1 -1	36.86989764584401285674

Note

The angle between two lines is always between 0 and 90 degrees. Pictures for both samples with the initial and one of the possible final positions with the largest angle are shown below.



Problem O. Optimum Server Location

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

The world's best IT company Oondex is coming to Lineland! After years of facing annoying "This service is not available in your area" error messages, Linelanders will finally be able to listen to the most popular music, watch fresh viral videos and use lots of other opportunities provided by Oondex's services.

Lineland can be seen as a real coordinate line. It has unusual network tariffs: connecting two servers d kilometers apart from each other with a network channel of throughput t Mbit/s costs $d \cdot t$ dollars.

In order to provide better user experience, Oondex is going to place n servers in Lineland. These servers will be performing regular data processing activities which requires intense pairwise network interaction between these servers. At the same time, these servers are going to serve external users using m special CDN servers (which are specialized content delivery servers) already present in Lineland.

Analysts of Oondex determined for each pair i, j ($1 \le i < j \le n$) the required throughput d_{ij} Mbit/sec between servers i and j, and also for each pair i, k ($1 \le i \le n$; $1 \le k \le m$) the required throughput c_{ik} Mbit/sec between server i and CDN server k.

Given the locations of CDN servers a_k ($1 \le k \le m$), determine the locations x_i ($1 \le i \le n$) such that the cost of placing servers into them is the minimum possible. Formally, determine x_i such that the cost value of $v = \sum\limits_{1 \le i < j \le n} |x_i - x_j| \cdot d_{ij} + \sum\limits_{\substack{1 \le i \le n \\ 1 \le k \le m}} |x_i - a_k| \cdot c_{ik}$ is the minimum possible. Multiple servers (both Oondex and CDN) may be located at the same point.

Input

The first line contains two integers n and m ($1 \le n, m \le 70$) — the number of Oondex servers to place and the number of existing CDN servers.

The second line contains m integers $a_1, a_2, \ldots, a_m$ ($0 \le a_k \le 10^6$) — the locations of existing CDN servers.

The i-th of the next n lines contains m integers $c_{i1}, c_{i2}, \ldots, c_{im}$ where c_{ik} ($0 \le c_{ik} \le 50$) is the throughput between i-th Oondex server and the k-th CDN server.

Finally, the i-th of the next n lines contains n integers $d_{i1}, d_{i2}, \ldots, d_{in}$ ($0 \le d_{ij} \le 50$; $d_{ij} = d_{ji}$; $d_{ii} = 0$) where d_{ij} is the throughput between j-th Oondex server and the i-th Oondex server.

Output

On the first line output the value v — the minimum possible cost of placing n Oondex servers.

On the second line output n integers $x_1, x_2, \ldots, x_n$ where x_i ($0 \le x_i \le 10^6$) — the coordinates at which the i-th Oondex server should be placed. It can be proven that an optimum answer satisfying these restrictions on x_i (range and integrality) exists.

Example

standard input	standard output
3 4 20 14 5 2 1 2 3 0 3 0 3 0 0 0 0 20 0 15 0 15 0 0 0 0 0	78 9 9 2