Problem A. Coins

Input file:	coins.in
Output file:	coins.out
Time limit:	2 seconds
Memory limit:	64 megabytes

You are given M positive integer values and one of these values is 1. You are also given an unlimited number of coins of each of these values. Consider the following problem: A certain amount of money S should be paid by a minimal number of coins with the given values.

It is known that this problem is solvable in some cases by the following greedy algorithm: Find the greatest value of a coin that is less than or equal to S, and then subtract it from S. Continue doing the same, until the value of S becomes zero. The number of coins, which is used by the algorithm to reduce S to zero seems to be the minimal number of coins needed at all.

In many cases the above assertion is true, but on some sets of values and for some S the greedy algorithm described above does not compute the optimal solution. For example, on a set of values $\{1, 2, 5, 7, 10\}$ and for $S = 14$, the greedy algorithm gives a solution with 3 coins ($14 = 10+2+2$), while the obvious minimal solution is with 2 coins ($14 = 7 + 7$).

A question arises — for which sets of values the greedy algorithm does not produce an optimal solution. Write a program that for a given set of coins' values should examine if there exists an amount S, which is representable by a smaller number of coins than the greedy algorithm says.

Input

On the first line of the input, the number M of different coins' values is given ($1 < M < 100$). On the second line of the input, the values $a_1, a_2, \ldots, a_M$ are given ($1 = a_1 < a_2 < \cdots < a_M \le 7000000$), separated by a single space. On the third line of the input two integers x and y are given ($0 < x < y \le 7000000$), separated by a space.

Output

The program should print on the first line of the output file a value S, $x \le S \le y$, for which the greedy algorithm fails to give the optimal solution. On the second line, the program should print the numbers $b_1, b_2, \ldots, b_M$ of coins (separated by a space), corresponding to the different values (in the same order as given in the input) to represent the amount of S, i.e. $S = a_1b_1 + a_2b_2 + \ldots + a_Mb_M$. The total number of used coins should be less than the number of coins obtained by the greedy algorithm. If there exists more than one solution, your program should print any of them.

The input data always guarantee the presence of at least one S for which the described greedy algorithm fails.

Example

coins.in	coins.out
5 1 2 5 7 10 1 100	14 0 0 0 2 0

Problem B. Game

Input file:	game.in
Output file:	game.out
Time limit:	2 seconds
Memory limit:	64 megabytes

Little Ivan likes to play games in his spare time. Unfortunately, he cannot always enjoy the company of his friends and sometimes he is a little bored when he is alone. Therefore, he makes up games, where he is the only player. He is especially proud of his last game and likes to tell you about it.

You are given two finite sequences of positive integers. The game consists of making consecutive moves. You are allowed to make the following move. You remove the last K_1 numbers ($K_1 \ge 1$) from the first sequence (possibly the whole sequence) and find their sum S_1 and the last K_2 numbers ($K_2 \ge 1$) from the second sequence (again you can remove the whole sequence) and find their sum S_2. Then you calculate the cost of the move to be $(S_1 - K_1)(S_2 - K_2)$. You continue to make moves until you remove all the numbers in both sequences. The total cost of the game is the sum of the costs of all moves. Your goal is to minimize this total cost. You are not allowed to leave one of the sequences empty, while the other is not.

As Ivan has told you the rules of the game, you realize that it is easily solvable with the help of a computer, so you decide to write a program that computes the minimum total cost of the game.

Input

Input data consist of three lines. The first line contains two space-separated integers, L_1 and L_2 ($1 \le L_1, L_2 \le 2000$), which denote the lengths of the two sequences. The second line contains L_1 space-separated integers, which are the elements of the first sequence. The third line contains L_2 space-separated integers, which are the elements of the second sequence. The elements of the sequences do not exceed 1000.

Output

Your program has to output one line that contains only one number — the minimum total cost of the game as described above.

Example

game.in	game.out
3 2 1 2 3 1 2	2

Problem C. Journey

Input file:	journey.in
Output file:	journey.out
Time limit:	2 seconds
Memory limit:	64 megabytes

There are n cities in Byteland (numbered from 1 to n), connected by bidirectional roads. The king of Byteland is not very generous, so there are only $n - 1$ roads, but they are connected in such a way that it is possible to travel from each city to any other city.

One day a traveller Byterider arrived in the city number k. He was planning to make a journey starting in the city k and visiting on his way cities $m_1, m_2, \ldots, m_j$ (not necessarily in this order) — the numbers m_i are all different and they are also different from k. Byterider — like every traveller — has only a limited amount of money, so he would like to visit all the cities that he has planned to visit using the shortest possible path (starting in the city k). A path is one road or a sequence of roads, where every next road begins in the city where the previous one ends. Help Byterider to determine the length of the shortest path for his journey.

Input

The first line of the input file contains two integers n and k separated by a single space ($2 \le n \le 50\,000$, $1 \le k \le n$), n is the number of cities in Byteland and k is the number of the first city on Byterider's path. Each of the following $n-1$ lines contain the description of one road each. Line $i+1$ contains three integers: a_i, b_i and d_i ($1 \le a_i, b_i \le n$, $1 \le d_i \le 1\,000$), a_i and b_i are the cities connected by the road, and d_i is the length of the road. Line $n+1$ contains one integer j — the number of cities which Byterider would like to visit ($1 \le j \le n-1$). The next line contains j different integers m_i separated by single spaces — the numbers of the cities that Byterider would like to visit ($1 \le m_i \le n$, $m_i \ne k$).

Output

Output the length of the shortest Byterider journey.

Example

journey.in	journey.out
4 2 1 2 1 4 2 2 2 3 3 2 1 3	5

Problem D. Puzzle

Input file:	puzzle.in
Output file:	puzzle.out
Time limit:	1 second
Memory limit:	64 megabytes

The king of Byteland has received a gift, a jigsaw puzzle. The puzzle consists of a board of size $n \times n$. The field in the i-th row and j-th column ($1 \le i, j \le n$) has coordinates (i, j) and contains a piece with the number $p(i, j)$, $1 \le p(i, j) \le n^2$. Each of the numbers $1 \ldots n^2$ appears on exactly one of the pieces.

To solve the puzzle, you have to put the pieces in order, so that for each $1 \le i, j \le n$ the field (i, j) contains the piece with number $j + (i-1)n$. The following moves are allowed to solve the puzzle:

- a cyclic shift of all pieces in a row by a certain number of fields to the right;
- a cyclic shift of all pieces in a column by a certain number of fields down.

The king of Byteland managed to solve his puzzle, but he is not sure if he would be able to solve it starting from a different initial configuration. Help him to solve this problem.

Input

In the first line of the input file there is one integer n — the size of the board side ($2 \le n \le 200$). The following n lines contain the description of the initial configuration. The line $i+1$ contains n integers $p(i,1), p(i,2), \ldots p(i,n)$ separated by single spaces.

Output

If there is no solution, the program should write to the output file only one line containing only one word "NO".

If a solution exists, the first line should contain one integer m — the number of moves leading to the solution of the puzzle. The number of moves in your solution must not exceed 400 000. The following m lines should contain the descriptions of the moves, one move per line. Each such line should consist of a letter 'R' (for shifting a row to the right) or 'C' (for shifting a column down), a space, and two integers: k and l separated by a space; $1 \le k \le n$, $1 \le l \le n-1$. A line containing "R k l" describes a cyclic shift of the k-th row by l fields to the right. Similarly a line containing "C k l" describes a cyclic shift of the k-th column by l fields down.

If there are several possible solutions, your program should output anyone of them.

Example

puzzle.in	puzzle.out
4 4 6 2 3 5 10 7 8 9 14 11 12 13 1 15 16	2 C 2 1 R 1 3

Problem E. Race

Input file:	race.in
Output file:	race.out
Time limit:	2 seconds
Memory limit:	64 megabytes

The mayor of Plovdiv decides to arrange a car race on the streets of Plovdiv to show that the streets of the city are really suitable for fast driving. He has to choose the route for the race, so this route should be as fast as possible. After talking with his advisors, he came up with the following constraints. The race must start and finish at the crossroad where the City Hall is. The only permitted turns along the track are left turns (to make a left turn means to change driving direction to the left in respect to the current forward direction by any angle greater than or equal to 0 and strictly less than 180 degrees). Moreover, among all routes, satisfying the above conditions, the chosen route should have the following property: the length of the shortest street from the route should be as long as possible.

The city of Plovdiv has N crossroads and M two-way streets connecting them. The crossroads are described with their two coordinates in the plane and numbered from 1 to N in the order they are given in the input. The City Hall is situated at

the crossroad with number 1. The streets are straight line segments starting at one crossroad and finishing at another. The length of a street is equal to the Euclidean distance between its two ends. A street is described by the numbers of its two ends. There is no more than one street between any two crossroads. The streets do not intersect themselves except at their endpoints.

Write a program to find which route, among all possible routes in the city, starts and finishes at crossroad 1 and at each crossroad this route goes either straightforward or makes a left turn, and the shortest street in this route is as long as possible. The route cannot pass twice in the same direction in the same street.

Input

The first line of the input file contains the two numbers N and M, $3 \le N \le 2000$, $5 \le M \le 25000$, separated by a space. Each of the next N lines contains coordinates X and Y of the given crossroads, separated by a space. These coordinates are integers from the interval $[-10000, 10000]$. Last M lines describe the streets. Each of these lines contains two crossroads' numbers, which are connected by a street.

Output

The output file should contain a description of the route. The first line should contain the count of the crossroads in the route (twice including crossroad 1 as the first and the last in the route). The next line should contain the numbers of the crossroads along the route (starting and ending with crossroad 1) in the proper order, separated by a space.

There is at least one solution of the task. If there is more than one solution, output an arbitrary one of them.

Example

race.in	race.out
5 6 1 0 2 1 1 1 0 1 1 2 1 2 2 5 1 4 5 4 2 3 4 3	5 1 2 5 4 1

Problem F. Sweets

Input file:	sweets.in
Output file:	sweets.out
Time limit:	1 second
Memory limit:	64 megabytes

John has got n jars with candies. Each of the jars contains a different kind of candies (i.e. candies from the same jar are of the same kind, and candies from different jars are of different kinds). The i-th jar contains m_i candies. John has decided to eat some of his candies. He would like to eat at least a of them but no more than b. The problem is that John can't decide how many candies and of what kinds he would like to eat. In how many ways can he do it?

Input

The first line of input contains three integers: n, a and b, separated by single spaces ($1 \le n \le 10$, $0 \le a \le b \le 10\,000\,000$). Each of the following n lines contains one integer. Line $i+1$ contains integer m_i — the amount of candies in the i-th jar ($0 \le m_i \le 1\,000\,000$).

Output

Let k be the number of different ways John can choose the candies to be eaten. The first and only line of output should contain one integer: $k \bmod 2004$ (i.e. the remainder of k divided by 2004).

Example

sweets.in	sweets.out
2 1 3 3 5	9

Problem G. Team Selection

Input file:	team.in
Output file:	team.out
Time limit:	2 seconds
Memory limit:	64 megabytes

The Interpeninsular Olympiad in Informatics is coming and the leaders of the Balkan Peninsula Team have to choose the best contestants on the Balkans. Fortunately, the leaders could choose the members of the team among N very good contestants, numbered from 1 to N ($3 \le N \le 500000$). In order to select the best contestants the leaders organized three competitions. Each of the N contestants took part in all three competitions and there were no two contestants with equal results on any of the competitions. We say that contestant is better than another contestant when is ranked before in all of the competitions. A contestant A is said to be excellent if no other contestant is better than A. The leaders of the Balkan Peninsula Team would like to know the number of excellent contestants.

Write a program, which for given N and the three competitions results, computes the number of excellent contestants.

Input

The input data are given as four lines. The first line contains the number N. The next three lines show the rankings for the three competitions. Each of these lines contains the identification numbers of the contestants, separated by single spaces, in the order of their ranking from first to last place.

Output

The output file should contain one line with a single number written on it: the number of the excellent.

Example

team.in	team.out
3 2 3 1 3 1 2 1 2 3	3
10 2 5 3 8 10 7 1 6 9 4 1 2 3 4 5 6 7 8 9 10 3 8 7 10 5 4 1 2 6 9	4

Problem H. Trips

Input file:	trips.in
Output file:	trips.out
Time limit:	5 seconds
Memory limit:	64 megabytes

In the forthcoming holiday season, a lot of people would like to go for an unforgettable travel. To mostly enjoy their journey, everyone wants to go with a group of friends. A travel agency offers several trips. A travel agency offers group trips, but for each trip, the size of the group is limited: the minimum and maximum number of persons are given. Every group can choose only one trip. Moreover, each trip can be chosen by only one group. The travel agency has asked you for help. They would like to organize as many trips as possible. Your task is to match groups of people and trips in such a way, that the maximum number of trips can be organized.

Input

The first line of input contains two integers: n and m separated by single space, $1 \le n \le 400000$, $1 \le m \le 400000$; n is the number of groups and m is the number of trips. The groups are numbered from 1 to n, and the trips are numbered from 1 to m. The following n lines contain group sizes, one per line. Line $i+1$ contains integer s_i — the size of the i-th group, $1 \le s_i \le 10^9$. The following m lines contain trip descriptions, one trip per line. Line $n+j+1$ contains two integers: l_j and u_j, separated by single space. l_j is the minimum, and u_j is the maximum size of a group for which the trip can be arranged, $1 \le l_j \le u_j \le 10^9$.

Output

The first line of output should contain one integer $k \ge 0$ — the maximum number of trips that can be arranged. The following k lines should contain the description of the matching. Each of these lines should contain a pair of integers separated by single space: the number of a group and the number of a trip. There can be many answers and your program may print anyone of them.

Example

trips.in	trips.out
5 4 5 4 6 9 4 2 15 6 6 20 50 2 8 7 20	3 2 1 3 4 4 2

Problem I. Two Sawmils

Input file:	two.in
Output file:	two.out
Time limit:	1 second
Memory limit:	64 megabytes

There are n old trees planted along a road that goes from the top of a hill to its bottom. Local government decided to cut them down. In order not to waste wood each tree should be transported to a sawmill.

Trees can be transported only in one direction: downwards. There is a sawmill at the lower end of the road. Two additional sawmills can be built along the road. You have to decide where to build them, as to minimize the cost of transportation. The transportation costs one cent per meter, per kilogram of wood.

Input

The first line of the input contains one integer n — the number of trees ($2 \le n \le 20\,000$). The trees are numbered $1, 2, \ldots, n$, starting from the top of the hill and going downwards. Each of the following n lines contains two positive integers separated by single space. Line $i+1$ contains: w_i — weight (in kilograms) of the i-th tree and d_i — distance (in meters) between trees number i and $i+1$, $1 \le w_i \le 10\,000$, $0 \le d_i \le 10\,000$. The last of these numbers, d_n, is the distance from the tree number n to the lower end of the road. It is guaranteed that the total cost of transporting all trees to the sawmill at the end of the road is less than $2\,000\,000\,000$ cents.

Output

The first and only line of output should contain one integer: the minimum cost of transportation.

Example

two.in	two.out
9 1 2 2 1 3 3 1 1 3 2 1 6 2 1 1 2 1 1	26