Problem A. Three Servers

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 mebibytes

There are three identical servers and n requests to process. The time to process each request is known: it takes t_i seconds to process the i-th request on any server.

You want to assign requests to servers in the most fair way. For each server, find its load: the total time required to process all requests assigned to this server. The assignment is the most fair if the difference between maximum and minimum load of the three servers is as small as possible.

Input

The first line contains an integer n ($1 \le n \le 400$), the number of requests. The second line contains the sequence of integer numbers $t_1, t_2, \ldots, t_n$ ($1 \le t_i \le 30$) where t_i is the time to process the i-th request.

Output

Print four lines. The first line must contain the smallest possible difference between the maximum and minimum load of servers. Each of the following three lines must contain the description of requests assigned to a server. A description must start with the number of assigned requests, followed by the indices of these requests in any order.

If there are several possible answers, print any one of them.

Examples

standard input	standard output
6 7 3 20 1 5 7	9 1 3 3 2 4 1 2 5 6
3 7 7 7	0 1 3 1 2 1 1

Problem B. Game on Bipartite Graph

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

This is an interactive problem.

You are given a bipartite graph with n vertices in the first part and m vertices in the second part. There are zero or more edges in the graph as well. Multiple edges are allowed. There is also a peg which is initially placed in one of the vertices of the first part. Two players are playing a game on this graph: on each turn, they move the peg along a graph edge. After the peg is moved along an edge, this edge disappears. The player who cannot make a move loses the game. The first player starts the game.

You are given the graph and the initial position of the peg. Your task is to write a program which will play this game as the first player. The jury's program will play as the second player, and it will win the game whenever it is possible. Your program should also win the game whenever it is possible. If it is impossible to win, your program should play until the game ends.

Your opponent's moves will be provided in interactive mode: after each of your moves, you will be given your opponent's next move. Please be sure to use the stream flushing operation after each of your moves to prevent output buffering. For example, you can use "`fflush(stdout)`" in C or C++, "`System.out.flush()`" in Java and "`flush(output)`" in Pascal.

Input

The first line contains four integers n, m, e and v ($1 \le n, m \le 50$, $0 \le e \le 2500$, $1 \le v \le n$): the number of vertices in the first part of the graph, the number of vertices in the second part, the number of edges and 1-based number of vertex in the first part where the peg is placed initially.

Next e lines describe edges. Each of these lines contains two numbers x and y ($1 \le x \le n$, $1 \le y \le m$) separated by a single space, denoting an edge between vertex x of the first part of the graph and vertex y of the second part.

After that, you will be given the opponent's moves. Each move is a 1-based integer v_i ($1 \le v_i \le n$), the number of the vertex in the first part of the graph where your opponent moved the peg.

Output

If you cannot make the next move, just print the phrase "`Player 2 wins`" on a separate line and terminate your solution gracefully. If a move is possible, print a line with a single integer: the number of vertex in the second part of the graph where you move the peg. If after your move, the opponent cannot make a move, additionally print the phrase "`Player 1 wins`" on a new line and terminate your solution gracefully. Do not forget to flush the output buffer after each of your moves.

Each time there are several possible moves, you can pick any one of them, provided that you still win in the end if you initially could.

Examples

standard input	standard output
3 2 5 3 1 1 1 2 2 1 2 2 3 2 1 2	 2 1 2 Player 1 wins
2 2 3 1 1 1 1 2 2 2	 1 Player 1 wins
2 2 6 1 2 2 2 2 1 1 1 2 2 1 2 2 2 1	 1 2 Player 2 wins

Problem C. Black and White Board

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 mebibytes

You are given a chessboard of size $n \times m$. The cells are painted black and white such that the cell $(1, 1)$ is white, and any two adjacent cells have different color. Some cells of the chessboard are banned.

Your task is to find a connected figure on the chessboard which does not contain banned cells and contains exactly w white cells and b black cells. Two cells are adjacent if they share a side.

Input

The first line contains three integers n, m and k ($1 \le n, m \le 10$, $0 \le k \le n \cdot m$): the number of rows of the chessboard, the number of columns of the chessboard and the number of banned cells.

The following k lines contain the coordinates of banned cells. Each of these lines contains a pair of integers x_i and y_i ($1 \le x_i \le n$, $1 \le y_i \le m$): the coordinates of a banned cell. All banned cells are distinct.

The last line contains two integers w and b ($0 \le w, b \le n \cdot m$, $w + b > 0$).

Output

If it is impossible to find a connected figure without banned cells with w white cells and b black cells, print a single line containing "`:-(`" without quotes.

Otherwise, print n lines containing m characters each which describe the figure. The lines must consist of characters "`O`", "`X`" and "`.`":

- character "`O`" means a white cell which is included in the desired figure,
- character "`X`" means a black cell which is included in the desired figure,
- character "`.`" means a cell which is not included in the desired figure.

If there are several possible answers, print any one of them.

Examples

standard input	standard output
3 4 2 2 2 2 3 3 3	.... X..O OXOX
8 10 1 1 5 19 9	....X...XOX. ..O...O... ..XO.OXO.. ..O.O.O.O. .OXOXOXOXO ..O.O.O.O. ....

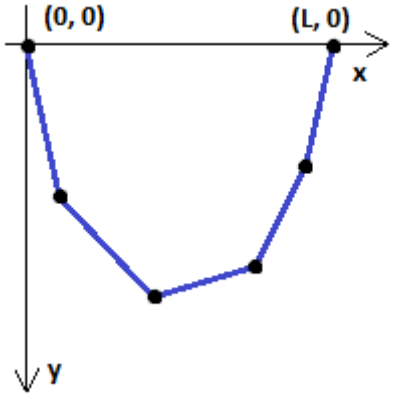
Problem D. Catenary

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

Everyone has seen something like this a lot of times:

Have you ever thought about the curve formed by a hanging chain? No, it is not a parabola, it is a graph of the hyperbolic cosine, called a *catenary*. In this problem, you have to draw a graph of some discrete analog of the catenary.

Suppose that you have several rods connected by their ends in a line in a certain order. The connections are non-stretchable and flexible, that is, each two successive rods can join at any angle. The rods are very thin, so they can be treated as line segments. The mass is uniformly distributed along each segment. Moreover, all the segments have the same density, so their masses are proportional to their lengths.



The ends of this chain are fixed at the points $(0, 0)$ and $(L, 0)$, and the segments form a polyline due to gravity. The force of gravity, as well as the y-axis, is directed downwards. Determine the form of the resulting polyline.

Input

The first line contains two integers n and L ($3 \le n \le 10$, $200 \le L \le 299$). The second line contains n integers $l_1, l_2, \ldots, l_n$ ($100 \le l_i \le 199$), the lengths of the segments from left to right. It is not allowed to change the order of the segments.

Output

Output $n - 1$ lines. Each line must contain two real numbers, presenting x and y-coordinates of the points of the polyline from left to right. Do not output the first and the last points $(0, 0)$ and $(L, 0)$.

Please print the numbers with at least three digits after the decimal point.

Examples

standard input	standard output
3 200 100 100 100	50.0000000000 86.6025403784 150.0000000000 86.6025403784
3 241 128 105 108	76.2967704600 102.7754971643 180.4447346423 89.4262815813

Problem E. Evacuation Plan

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 mebibytes

The main office of the Berland Bruteforce Corporation consists of n rooms, connected by $n-1$ passages. There is exactly one way to travel between any pair of rooms using these passages (seeing this phrase, one who is familiar with the graph theory immediately recognizes an undirected tree). Let's assume that the rooms are numbered with consecutive integers from 1 to n. There are e_i employees working at room i. For each passage, its length (in meters) is known.

The recent earthquakes have convinced the board of BBC to create an evacuation plan for the employees. In a plan, there should be a single evacuation point, which may be located either at some room or at any place in a passage. In the latter case, a new room is created. This new room divides the passage of length L into two passages of lengths L_1 and L_2 ($L_1 + L_2 = L$).

Once the evacuation point is set, the evacuation happens as follows. At the time 0 (the beginning of evacuation), all employees simultaneously try to start moving towards the evacuation point. It takes exactly s seconds for a person to walk 1 meter. Passages have limited capacity c, which is the same for all passages: at most c persons can enter the passage at a time step. Let us clarify this a bit. Consider some passage. It is clear that people will move through it only in one of the two possible directions. At time 0, at most c persons can enter the passage. Then, nobody can enter the passage until time 1, when again at most c persons can enter the passage, an so on. There is no limit on the number of persons that can be in a room or in a passage at the same moment of time.

The BBC board is interested in such a plan that minimizes the time of the evacuation: the time when every person reaches the evacuation point. Help them to create it.

Input

The first line contains three integers n, c and s ($1 \le n \le 10^5$, $1 \le c \le 10^4$, $1 \le s \le 100$): the number of rooms, the capacity of each passage and the time needed to walk 1 meter. The second line contains n integers $e_1, e_2, \ldots, e_n$ ($1 \le e_i \le 10^6$). Each of the next $n-1$ lines contains three integers u, v and d ($1 \le u, v \le n$, $1 \le d \le 10^4$), meaning that there is a passage of length d meters between rooms u and v.

Output

Print the optimal location of the evacuation point. If you are going to place the point in an existing room, print its number. Otherwise, print three numbers u, v and x, meaning that the evacuation point must be placed in the passage connecting rooms u and v at the distance of x meters from u. The number x must belong to the interval $(0, l)$ where l is the length of the passage between u and v.

Your answer will be considered correct if the evacuation time in your plan has an absolute or relative error of at most 10^{-9} from an optimal one.

Examples

standard input	standard output
2 2 1 5 5 1 2 3	1 2 1.500000000000
2 2 1 5 10 1 2 3	1 2 2.500000000000
3 2 10 8 6 8 1 2 10 2 3 10	2
4 3 1 3 8 4 7 1 2 2 2 3 1 2 4 5	2 4 1.500000000000

Note

Let's go through the evacuation in the fourth sample step-by-step.

- $t = 0$:
  - 3 persons enter the passage $(1, 2)$ from room 1, they'll arrive at room 2 at $t = 2$.
  - 3 persons enter the passage $(3, 2)$ from room 3, they'll arrive at room 2 at $t = 1$.
  - 3 persons enter the passage $(4, 2)$ from room 4, they'll arrive at evacuation point at $t = 3.5$.
  - 3 persons enter the passage $(2, 4)$ from room 2, they'll arrive at evacuation point at $t = 1.5$.
- $t = 1$:
  - 1 person enters the passage $(3, 2)$ from room 3, she'll arrive at room 2 at $t = 2$.
  - 3 persons from room 4 arrive at room 2.
  - 3 persons enter the passage $(4, 2)$ from room 4, they'll arrive at evacuation point at $t = 4.5$.
  - 3 persons enter the passage $(2, 4)$ from room 2, they'll arrive at evacuation point at $t = 2.5$.
- $t = 2$:
  - 3 persons from room 1 and 1 person from room 3 arrive at room 2.
  - 1 person enters the passage $(4, 2)$ from room 4, she'll arrive at evacuation point at $t = 5.5$.
  - 3 persons enter the passage $(2, 4)$ from room 2, they'll arrive at evacuation point at $t = 3.5$.
- $t = 3$:
  - 3 persons enter the passage $(2, 4)$ from room 2, they'll arrive at evacuation point at $t = 4.5$.
- $t = 4$:
  - 3 persons enter the passage $(2, 4)$ from room 2, they'll arrive at evacuation point at $t = 5.5$.

Evacuation finishes in 5.5 seconds.

Problem F. Empty Vessels

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

There are n empty vessels on a river bank. The i-th vessel can contain at most a_i liters of water. Almighty Ivan needs to fill some vessel with exactly A liters of water and take it home. To do so, he can perform the following operations:

1. fill some vessel with water from the river until it is full,
2. pour out water from some vessel to the river so it becomes empty,
3. pour water from the i-th vessel to the j-th vessel until the i-th one is empty or the j-th one is full.

Now Almighty Ivan needs your help. You should provide him with the list of operations which leads to the situation when one of the vessels contains exactly A liters of water. If there is no way to do so, print -1.

Please note that you don't have to minimize the number of operations in this list, but the number of operations should not exceed 10^6.

Input

The first line contains integers n and A ($1 \le n \le 10$, $1 \le A \le 5$) — the number of vessels and the required number of liters. The next line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2 \cdot 10^4$) where a_i is the volume of the i-th vessel.

Output

If the answer doesn't exist, print -1.

Otherwise, on the first line, print one integer m: the number of operations in the list. Then print m lines: the descriptions of the operations.

The description of each operation must start by an integer specifying its type (1, 2 or 3 as they are listed in the statement). For the first and second type, it must be followed by one integer number k ($1 \le k \le n$) which is the index of the vessel taking part in this operation. For the third type, it must be followed by two integer numbers i and j ($1 \le i, j \le n$, $i \ne j$) which mean that Ivan should pour water from the i-th vessel to the j-th one until one of them is full or empty.

Please check the samples for better understanding.

Examples

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

Problem G. Maximum Product

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

Find the number from the range $[a, b]$ which has the maximum product of the digits.

Input

The first line contains two positive integers a and b ($1 \le a \le b \le 10^{18}$): the left and the right ends of the range.

Output

Print the number with the maximum product of the digits from the range $[a, b]$. If there are several possible answers, print any one of them.

Examples

standard input	standard output
1 10	9
51 62	59

Problem H. Biathlon 2.0

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

The Biathlon 2.0 World Championship is held in Petrozavodsk in 2016.

Biathlon 2.0 has different rules than the classic one. The racing track consists of n segments. The i-th segment consists of a_i kilometers to travel and b_i targets to hit. To pass a segment, an athlete must travel all a_i kilometers and hit all b_i targets. Athletes pass segments in the order from 1 to n. Each athlete must carry a rifle with him. It is allowed to change rifles between segments.

The national team of Berland has m different rifle types at its disposal. Each rifle type is described by two numbers: c_i and d_i, where c_i is the number of seconds needed to travel one kilometer with a rifle of this type and d_i is the number of seconds needed to hit one target with a rifle of this type. The team has an unlimited supply of rifles of any type.

Your task is to ensure Berland victory and find a rifle for each track segment to minimize the number of seconds needed to pass the segment.

Input

The first line contains one integer n ($1 \le n \le 5 \cdot 10^5$), the number of segments in the track. Each of the next n lines contains two integers a_i and b_i ($0 \le a_i, b_i \le 10^9$, $a_i + b_i > 0$), the number of kilometers and targets on the i-th track segment.

The next line contains one integer m ($1 \le m \le 5 \cdot 10^5$), the number of rifle types used by the national team of Berland.

The next m lines contain two integers c_i and d_i ($1 \le c_i, d_i \le 10^9$), the number of seconds needed to travel one kilometer with the i-th rifle type and the number of seconds needed to hit one target with the i-th rifle type.

Output

Print n space-separated integers. The i-th integer must be the number of seconds needed to pass the i-th segment if an optimal rifle is used.

Examples

standard input	standard output
3 1 4 4 1 3 3 3 1 3 3 1 2 2	7 7 12
1 1000000000 1000000000 1 1000000000 1000000000	2000000000000000000

Problem I. Archaeological Research

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

Professor Tupids has found a mysterious manuscript during his recent archaeological expedition.

The manuscript, in fact, looks like a series of symbols with meaning yet to be discovered. In order to simplify studying of the manuscript, professor Tupids created an alphabet of all symbols occurred (let's denote their number as c) and replaced each symbol with its position in the alphabet (positions are numbered from one). So, the manuscript became represented as a list of n integers from the segment $[1; c]$.

Professor's intuition told him that the key to understanding the manuscript is to find some regularities in locations of the symbols. So he wrote down a huge table with n rows and c columns, where cell at i-th row and j-th column contained the position of the next occurrence of symbol number j after the position i (the cell was left empty if there were no appropriate occurrences).

But then a disaster happened: a fire in the laboratory completely destroyed the manuscript! Fortunately, the table built by professor was salvaged, though it was damaged to some extent. Not only some of the cells were lost in the fire, but, thanks to the careless assistants, cells in each of the rows were reordered arbitrarily!

Professor Tupids doesn't want to lose face in the scientific community, so he asks you to help him with restoring the original manuscript, given the remaining information from the table. As there may be infinitely many different solutions (even the size c of the alphabet was lost!), professor wants you to restore the lexicographically smallest solution. It is not guaranteed that the solution exists, though, as the table could have been completely spoiled by the assistants.

Input

The first line contains a single integer n ($1 \le n \le 3 \cdot 10^5$), the length of the original manuscript. After that, n lines follow, the i-th of which contains an integer c_i ($0 \le c_i \le n - i$) followed by c_i distinct integers from the segment $[i + 1; n]$: the contents of the survived non-empty cells of the i-th row of the table in arbitrary order.

It is guaranteed that the sum of all c_i ($1 \le i \le n$) does not exceed $3 \cdot 10^5$.

Output

If the solution exists, print a single line with n positive integers: the representation of the lexicographically smallest manuscript. Otherwise, print "`No solution`" (without quotes).

Examples

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

Problem J. Sockets

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

Valera has only one electrical socket in his flat. He also has m devices which require electricity to work. He's got n plug multipliers to plug the devices, the i-th plug multiplier has a_i sockets.

A device or plug multiplier is supplied with electricity if it is either plugged into the electrical socket, or if it is plugged into some plug multiplier which is supplied with electricity.

For each device j, Valera knows the safety value b_j which is the maximum number of plug multipliers on the path between the device and the electrical socket in his flat. For example, if $b_j = 0$, the device should be directly plugged into the socket in his flat.

What is the maximum number of devices Valera could supply with electricity simultaneously? Note that all devices and plug multipliers take one socket to plug, and that he can use each socket to plug either one device or one plug multiplier.

Input

The first line contains two space-separated integers n and m ($1 \le n, m \le 2 \cdot 10^5$), the number of plug multipliers and the number of devices correspondingly.

The second line contains n space-separated integers a_1, a_2, ..., a_n ($2 \le a_i \le 2 \cdot 10^5$). Here, the number a_i stands for the number of sockets on the i-th plug multiplier.

The third line contains m space-separated integers b_1, b_2, ..., b_m ($0 \le b_j \le 2 \cdot 10^5$). Here, the number b_j stands for the safety value of the j-th device.

Output

Print a single integer: the maximum number of devices that could be supplied with electricity simultaneously.

Examples

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

Problem K. Toll Roads

Input file:	*standard input*
Output file:	*standard output*
Time limit:	10 seconds
Memory limit:	256 mebibytes

There are n cities in the country of Flatland, $n-1$ bidirectional roads connect some pairs of Flatland's cities. It is possible to get from any city to any other city of the country using roads. In this problem we will refer to the smallest set of roads needed to travel from a to b as a *simple path*.

The government of Flatland has recently introduced payment tolls on all roads. Using one road connecting two cities costs one ruble. That means that every time when you travel from one city to another one, you have to pay as many rubles as the number of roads you pass during your journey.

Many people got upset because of this decision, especially people who travel long distances. Political opponents of the current government claim that the maximal cost of a simple path in Flatland is very high, namely *cost* rubles. The government has decided to support people and calm down the opposition. They want to reduce the maximal cost of a simple path in the country as much as possible. In other words, they want the number *cost* reported by the opposition to be the lowest possible. The government will allow to remove tolls from at most k roads in order to achieve that. If there are multiple ways to do that, they want to minimize the number of roads that will become free.

Like any other government, Flatland's one is quite inflexible. They additionally require that it should be possible to represent the set of roads which will become free as a simple path between some cities x and y. Your task is to help the government.

Input

The first line contains two integers n and k ($1 \le k < n \le 5000$), the number of cities in Flatland and the maximum number of roads to become toll-free, respectively. Each of the next $n-1$ lines contains two integers u_i and v_i, meaning that there is a road connecting cities u_i and v_i ($0 \le u_i, v_i < n$).

Output

On the first line of the output, print one integer: the minimum *cost* of the most expensive simple path in Flatland that the government can achieve ($0 \le cost < n-1$). On the second line, print the minimum number of roads t that must be made free to achieve that *cost* ($0 \le t \le k$). If $t > 0$, print two space-separated integers on the third line: the numbers of cities x and y such that all roads on the simple path between them must be made free ($0 \le x, y < n$, the simple path between x and y must contain exactly t roads).

Examples

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