Astana Roads

The ~~Dictator~~ President of Kazakhstan is a talented individual who single-handedly designed the road network of Astana. It is perfect in every way you can imagine. There are no traffic jams at all. Never. Believe me.

The president is now starting an ambitious project: building the road network of a whole new city, New Astana. He already created the plan of an *even more* perfect road network. The road network will have N city squares, numbered from 0 to $N-1$ (inclusive), connected by $N-1$ road segments. The road segments are bidirectional and it is possible to travel between any pair of squares using the roads.

The Ministry of Construction must carry out the plan and build the road network specified by the president as quickly as possible. For this, they first build one of the city squares (they are free to choose any of the N options), and then each year they expand the current road network as follows: from **every** city square built earlier, they can construct a new road segment and build a new city square at the end of the road. It is allowed to skip a city square and not construct a new road from there. Note that it is not allowed to connect already existing city squares with a new road.

Your task is to find the minimum years required to build the complete road network.

Input

The first line of the input contains the integer N ($2 \leq N \leq 100\,000$), the number of city squares.

Each of the next N lines describes the road segments connecting a city square to other city squares. Line i (where $0 \leq i < N$) begins with a positive integer M_i, followed by M_i integers $A_{i,k}$ ($0 \leq A_{i,k} < N$ for each $k = 0 \ldots M_i - 1$). This means that there is a road segment connecting city square i to city square $A_{i,k}$.

Output

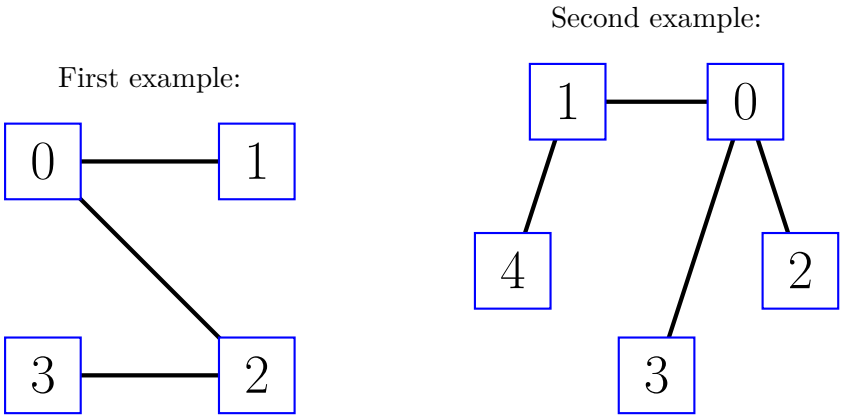
Print a single line containing a positive integer: the minimum years required to build the road network.

Examples

input	output
4 2 1 2 1 0 2 0 3 1 2	2

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

Explanation



In the first sample case, one optimal way to build the network is as follows: we first build square 2, then in the first year we build a new road from square 2 ending at square 0. In the second year, we build a new road from square 0 ending at square 1, and a new road from square 2 ending at square 3, completing the construction. It can be proved that it is impossible to construct the road network in just one year.

Borat And Squares

Borat has three papers of square shapes with side lengths A, B, and C. He wants to form a rectangle using these papers, such that he places them on a flat surface next to each other without any gap or any overlap between the papers. Can he do this?

Input

The first line contains three integers A, B, and C - the side lengths of the squares ($1 \leq A, B, C \leq 100$).

Output

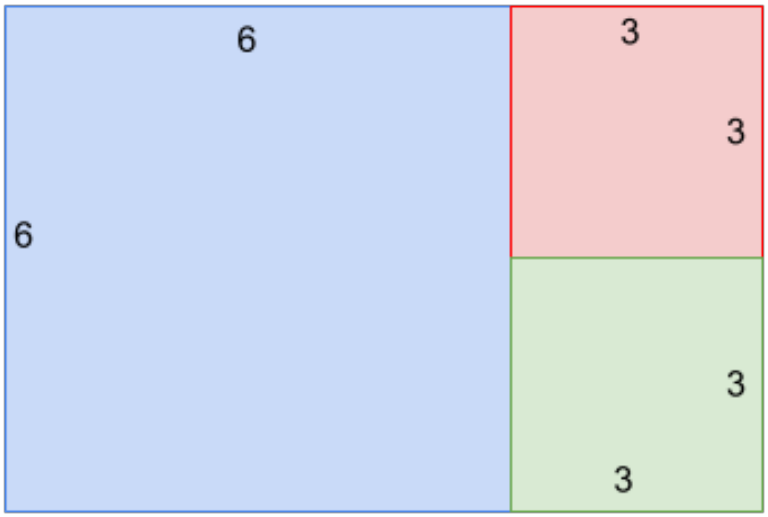
Print a single line containing the string `YES` if it is possible to form a rectangle using the three squares. Otherwise, print `NO`.

Examples

input	output
3 6 3	YES
4 4 4	YES
3 1 2	NO

Explanation

In the first sample case, $A = 3$, $B = 6$, and $C = 3$. Borat can create a 6×9 rectangle as shown below.



Cheese!

As university delegations are arriving at the venue of the ICPC World Finals, a group photo session is taking place. Your delegation consists of N people numbered from 1 to N, and you must stand in a straight line next to each other to shoot the photo.

But there is an issue: you cannot decide the order in which the members should stand in the line so that the photo looks as great as possible. After arguing for a while, you finally ask the photographer to quickly take photos of your delegation in every possible order in which they can stand up in a line. The photographer is not happy with this and wants to ensure that the shooting session won't take too much time: after taking a photo, exactly two delegation members will quickly swap places with each other, and then they take the next photo. This process is repeated until there is at least one photo of every possible ordering of the members.

Your task is to find a sequence of orderings of the N delegation members for the photoshoot session, such that

- exactly two members swap places between every pair of consecutive photos, and
- the number of photos taken (that is, the length of the sequence) is minimal.

Input

The first line of the input contains the integer N ($2 \leq N \leq 9$), the number of delegation members.

Output

Print a line containing a positive integer M, the minimum number of photos that must be taken to capture each possible ordering. Then print M additional lines, each containing exactly N distinct integers between 1 and N (inclusive). Line i should contain the order of the delegation members for the i-th photo.

Examples

input	output
2	2 1 2 2 1

Deer Dance

The glorious nations of Kazakhstan and Hungary have much more in common than you would think! One such thing is the popularity of the venerable mythological creature *Csodaszarvas* among common folks. In Astana, there is a large statue depicting this deer-like creature. For the ceremonies of the Nomad Games 2024, the organizers want to decorate the antlers of the statue with red and green decorative balls.

We model the statue's antlers as a rooted tree graph of N nodes numbered from 0 to $N-1$. The root of the graph is node 0, and node P_i is the parent node of node i for every $0 < i < N$.

The organizers have an infinite number of red and green balls available. They want to hang some balls from the nodes of the graph according to the following rules:

- They can put any number of balls (including 0) to each node.
- If they hang at least 1 red ball from some node i, then they are not allowed to put any green balls to nodes $i, P_i, P_{P_i}, \ldots, 0$. In other words, neither i nor any of its ancestors can have green balls hanging from them.
- If node i is a leaf node, consider the set of nodes $i, P_i, P_{P_i}, \ldots, 0$ (i.e., the path from the leaf to the root). They must hang exactly R red and G green balls to the nodes in this set **in total**.

Professor Tibor von Minerale is visiting Kazakhstan to watch the Nomad Games. When he sees the decorated statue, he wonders: what is the total number of ways the organizers could have decorated it?

Input

The first line of the input contains the integer N ($1 \leq N \leq 300$), the number of nodes in the tree graph.

The second line contains $N-1$ integers P_i ($0 \leq P_i < i$), the parent of node i for each i from 1 to $N-1$.

The third line contains two integers R and G ($0 \leq R, G \leq 10^9$).

Output

Print one line containing the number of ways to decorate the antlers according to the rules. As this number can be large, output it modulo $10^9 + 7$.

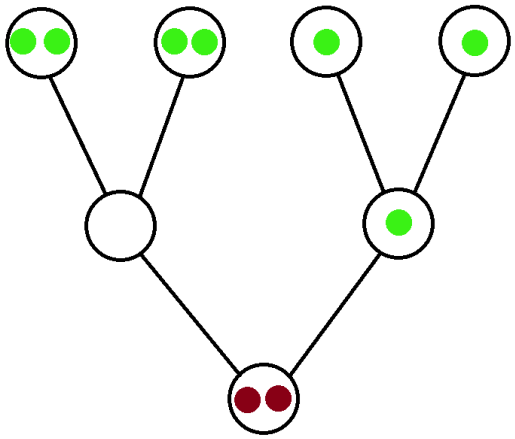
Examples

input	output
7 0 0 1 1 2 2 2 2	11

input	output
3 0 0 5 0	6
1 8 9	0

Explanation

In the first sample case, one possible way to decorate the antlers is shown in the following figure.



Elevators of SAADness

You are staying at Hotel SAAD in Astana. The hotel provides high-quality accommodation with one minor flaw: the building has many floors, but just 3 elevators to serve them. Suppose the hotel has N floors, where N is an **odd** integer. The floors are numbered from 0 to $N-1$. Initially all 3 elevators are on floor $\frac{N-1}{2}$.

To make things worse, the elevators use a strange energy saving-protocol: if one of the elevators travels some number of floors up or down, then another elevator must simultaneously travel the same number of floors in the **other** direction. More formally, whenever an elevator travels from floor x to floor y (including the scenario where $x = y$), that is, the floor number of the elevator changes by $y - x$, then another elevator must simultaneously change its floor number by $x - y$. Following this, both elevators must be on floors with indices between 0 and $N-1$.

For example, suppose that the building has $N = 5$ floors and the elevators are currently on floors 2, 1, and 3, respectively. If the first elevator wants to go from its current floor 2 to floor 3, then at the same time either the second elevator must go from floor 1 to floor 0, or the third elevator from floor 3 to floor 2. However, if the first elevator goes from floor 2 to floor 0, the second elevator must go from floor 1 to floor 3, as the third elevator cannot move two floors up from floor 3.

If a guest wants to go from one floor to another, the following steps are performed:

1. the control system selects one of the elevators to serve the request, then
2. the selected elevator travels to the floor of the guest with the help of one of the two other elevators, and finally
3. the selected elevator travels to the destined floor with the help of one of the two other elevators.

It is allowed to use different elevators to support the selected elevator for steps 2 and 3. However, during performing one of the steps, it is not allowed to change the supporting elevator.

There are Q consecutive requests, each described by two integers X_i and Y_i, meaning that a guest wants to go from floor x_i to floor y_i. The requests must be served one by one in the given order. It can be proved that there is always at least one way to serve the requests. You are wondering what is the minimum number of floors that the selected elevators travel in total while serving the requests if they are scheduled optimally.

Input

The first line of the input contains two integers N ($3 \leq N < 64$, N is odd) and Q ($1 \leq Q \leq 200$), the number of floors and the number of requests, respectively.

Each of the next Q lines contains two integers X_i and Y_i ($0 \leq X_i, Y_i < N$), describing the requests in the order they must be served.

Output

Print a line containing a single integer, the minimum number of floors that the selected elevators travel.

Examples

input	output
5 2 1 2 1 0	3

Explanation

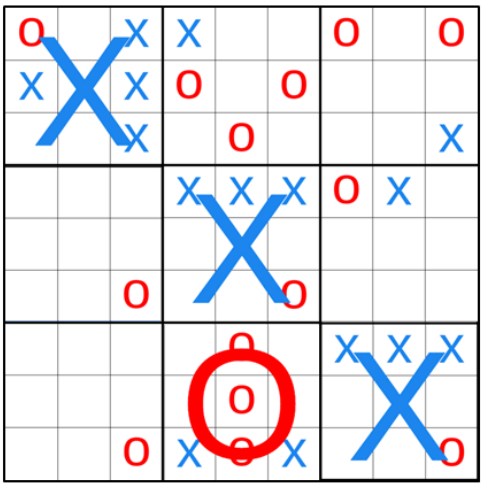
There are $N = 5$ floors and two requests. An optimal way to serve them is as follows:

- Initially, all three elevators are on floor 2. Select the first elevator to serve the first request. Move the elevator from floor 2 to floor 1 while moving the second elevator from floor 2 to floor 3, then move the elevator from floor 1 to floor 3 while moving the third elevator from floor 2 to floor 1. The first elevator traveled 2 floors total.
- After serving the first request, the elevators are on floors 2, 3, and 1. Select the third elevator for the second request. It is already on floor 1, so it can pick up the guest and travel to floor 0 while the first elevator moves from floor 2 to floor 3. The third elevator traveled 1 floor total.

We see that the selected elevators traveled $2 + 1 = 3$ floors total. It can be proved that it is impossible to serve the requests by traveling fewer total floors.

Frenetic-tac-toe

Xeniya and Olga play a version of the tic-tac-toe game that they call frenetic-tac-toe. The game is played on a 3×3 grid of smaller 3×3 tic-tac-toe boards, making 9 small boards in total. Xeniya and Olga take turns placing their mark (X or O, respectively) on one of the small boards. They can choose freely to make a move on any of the small boards. Regular tic-tac-toe rules apply within each small board, so a player wins a small board by placing three marks (X or O) in a row (horizontally, vertically, or diagonally). If a small board is won by someone, it is forbidden to place further marks on that board. A player wins the entire game by winning three small boards in a row (horizontally, vertically, or diagonally) on the larger 3×3 grid of boards. A small board and the larger game might also end in a draw if the players cannot place a mark anvwhere. and there is no winner.



A game of frenetic-tac-toe won by Xeniya.

You see the current state of the game. You wonder whether the game is over already and what the outcome is. The following options are possible:

- Xeniya won the game by winning three small boards in a row (horizontally, vertically, or diagonally)
- Olga won the game by winning three small boards in a row (horizontally, vertically, or diagonally)
- The game ended in a *draw*, i.e. there is no winner, but it is impossible to place a mark anywhere (all the small boards are either won by someone or full).
- The game is *unfinished*, i.e. there is no winner yet and it is possible to place a mark somewhere.

Input

The input contains 9 lines with 9 characters each, depicting the board. An uppercase letter `X` denotes a cell marked by Xeniya, an uppercase letter `O` denotes a cell marked by Olga, and a `.` denotes a free cell. It is guaranteed that there exists a valid sequence of moves leading to the given state (for example, there cannot be two winners within one small board).

Output

Print a single line containing one of the following strings depending on the outcome of the game:

- `XENIYA` if Xeniya won.
- `OLGA` if Olga won.
- `DRAW` if the game ended in a draw.
- `UNFINISHED` if the game is unfinished.

Examples

input	output
O.XX..O.O X.XO.O... ..X.O...X ...XXXOX. ....O..O... ....O.XXX ....O.... ..OXOX..O	XENIYA
XXX..OXOX ....O.OXO ...O..OXO X..X....O XO.OOOXXX X.......O .O.X..XOO .O..X.OXX .O...XXXO	DRAW
XXX..OXOX ....O.OX. ...O..OXO X..X....O XO.OOOXXX X.......O .O.X..XOO .O..X.OXX .O...XXXO	UNFINISHED

Explanation

The first sample case corresponds to the picture in the statement.

In the second sample case, all the small boards are either won by someone or full, but the larger game has no winner, so the game ended in a draw.

In the third sample case, there is one empty cell in the upper-right small board where the next player can place her mark. Although it can be seen in advance that the game will end in a draw, it is still unfinished.

Get Some Rest

The ICPC World Finals is a huge event with many participants. The organizers have already booked some hotels for all the teams, coaches, guests, and other attendees. They assigned everyone to one of the hotels, but it turns out that this initial arrangement put more people in some hotels than their maximum capacity, and also some other hotels still have a bunch of available beds left. Now they will try to fix the issue somehow.

There are N hotels numbered from 0 to $N-1$, in ascending order of quality (hotel $N-1$ is the best hotel, and hotel 0 is the worst). The initial state of hotel i ($0 \le i < N$) is represented by an integer A_i:

- if A_i is positive, then there are A_i free beds in hotel i;
- if A_i is negative, then there are $-A_i$ people in hotel i over the hotel's max capacity;
- if A_i is zero, then the hotel is full (no free beds nor people without accommodation in the hotel).

The organizers will move the free beds to other hotels because they don't want to upset people by moving them away from their friends and delegations. They also want to offer people better beds than they'd otherwise get, so to each hotel i ($0 < i < N$) they assigned a worse quality hotel $B_i < i$, and they will move free beds from hotel i to hotel B_i. There is no hotel worse than hotel 0, so if there are free beds in hotel 0 that need to be moved, they will move them to one specific better hotel $B_0 > 0$.

To move the beds, the organizers perform QuickAssign™, a famous algorithm by Professor Tibor von Minerale. In case you cannot remember, here is a description of the algorithm. In one step, QuickAssign™ will

1. select the hotel with the largest number of free beds (if there are more with the same number, it selects the one with the lowest index), and
2. if the hotel selected for this step is hotel x, then send every free bed from hotel x to hotel B_x, and assign as many beds as possible to people without accommodation in hotel B_x.

The process ends once there are no more free beds, or there are no people left without accommodation.

Despite the professor's brilliancy, QuickAssign™ is not perfect. Your task is to determine whether it solves the issue successfully, and if so, then the number of steps it takes.

Input

The first line of the input contains the integer N ($2 \le N \le 200\,000$), the number of hotels.

The second line contains N integers A_i ($-10^9 \le A_i \le 10^9$), representing the initial state of each hotel.

The third line contains N integers B_i ($0 < B_0 < N$ and $0 \le B_i < i$ for each $i = 1 \dots N$), the hotel indices where free beds will be sent.

Output

If the algorithm never terminates, or there are people left without accommodation on its termination, print a single line containing -1. Otherwise (i.e., if the algorithm successfully solves the issue), print a line containing the number of steps until termination.

Examples

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

Explanation

In the first sample case, the algorithm moves 2 beds from hotel 2 to hotel 1, and then from hotel 1 to hotel 0 over the first two steps. After that, the states of the hotels are $[-1, 0, 0, 1]$. After three more steps, the last person finally receives a bed and the algorithm terminates.

In the second sample case, there are no free beds to move after 5 steps, so the algorithm terminates without success.

In the third sample case, the states of the hotels after each step are as follows:

- after the first step: $[1, 0, 2, -1]$;
- after the second step: $[1, 2, 0, -1]$;
- after the third step: $[3, 0, 0, -1]$; and
- after the fourth step: $[0, 0, 0, 2]$.

At this point there are no people left without a bed, so the algorithm terminates with success.

Hot and Cold

The ICPC is the most prestigious programming contest for college students. As you can expect, every student wants to maximize their performance at the World Finals of the competition. Also, they can easily get whiny and insufferable if the conditions in the contest hall are not optimal for them.

In this problem, we are focusing on the temperature of the contest hall and would like to set it optimally. The temperature can be set to an arbitrary numeric value (that is **not** necessarily an integer).

Suppose that there are N contestants attending the World Finals, numbered from 0 to $N-1$ (inclusive). Each contestant is described using four integers A_i, B_i, C_i, and L_i, determining their satisfaction value: if the room temperature is set to x, then the satisfaction value of contestant i is $A_i \cdot x^2 + B_i \cdot x + C_i$. However, if x is either negative or greater than the maximum tolerance L_i of contestant i, then the contestant refuses to compete and leaves.

We want to set the temperature so that the sum of the satisfaction values of the contestants is maximized. Your task is to compute this maximum value. (If a contestant leaves, their contribution to the total satisfaction value is 0.)

Input

The first line of the input contains an integer N ($1 \le N \le 100\,000$), the number of contestants. After this, there are N lines, each containing four integers A_i, B_i, C_i, and L_i ($-10^9 \le A_i, B_i, C_i \le 10^9$, $1 \le L_i \le 100\,000$), describing a contestant. The function $A_i \cdot x^2 + B_i \cdot x + C_i$ is positive over all values of x between 0 and L_i, inclusive.

Output

Print a single number: the maximum possible sum of all satisfaction values. The answer will be accepted if it has an absolute or relative error of at most 10^{-5}.

Examples

input	output
2 1 -6 10 4 1 -6 10 7	20.00000000
2 -3 20 3 5 -1 0 2 1	36.33333333
1 1000000 2 3 10000	100000000020003.00000000

Explanation

In the first sample case, the optimal room temperature is $x = 0$. In this case, the satisfaction value of both contestants is 10, so the total satisfaction is 20. It can be proved that the sum is less than 20 for any other value of x.

In the second sample case, the optimal temperature is $x = \frac{10}{3}$, at which the satisfaction value of contestant 0 is $\frac{109}{3}$, and contestant 1 leaves, contributing 0 satisfaction.

Inboxes

The ICPC organization has many directors, each responsible for some aspect of the ICPC events, ranging from the regional to the global scale. However, as the organization expanded over the decades, the number of directors grew ridiculously large. Nowadays, when teams or coaches have some questions or issues that they want to sort out with the ICPC office, they don't know who to reach out to: they usually send an email to the inbox of multiple directors at once. Unfortunately, the directors receiving these messages are usually not responsible for the issue at hand, or they are just too ~~lazy~~ busy to deal with the issue. Therefore, they sometimes simply forward their messages to some other director.

Suppose the ICPC office has N directors numbered from 0 to $N-1$. Initially, the inbox of director p contains A_p messages, each read by the director previously. Consider Q consecutive days. On each of these days, exactly one of two possible events happens. The event on day i ($0 \le i < Q$) is described by four integers T_i, L_i, R_i, and X_i, where T_i is the type of the event (either 0 or 1). In the following, the $\oplus$ symbol denotes the binary XOR operation.

- If $T_i = 0$: director p, for every p between L_i and R_i (inclusive), receives X_i new **unread** messages to their inbox from teams, coaches, etc.
- If $T_i = 1$: director p, for every p between L_i and R_i (inclusive), forwards all of their **read** messages to director $p \oplus X_i$, and deletes these messages from their own inbox. The messages arrive in the inbox of director $p \oplus X_i$ as (new) unread messages. Note that directors do not forward these newly received messages on this day.

At the end of the day, every director reads their new messages, that is, every unread message turns into a read message before the next day starts.

After these Q days, the directors realize that the next ICPC World Finals is way too close and they must focus on replying to the emails from now. Director p ($0 \le p < N$) needs S_p seconds to answer a single email. Find the **total** time that the directors spend answering the emails.

Input

The first line of the input contains two integers N ($1 \le N \le 2^{17}$, N is a power of 2) and Q ($1 \le Q \le 100\,000$), the number of directors and the number of days, respectively.

The second line contains the N integers A_p ($0 \le A_p \le 10\,000$), the number of messages in the inboxes at the start.

Each of the next Q lines describes the event of the day. The i-th line contains four integers T_i ($T_i = 0$ or $T_i = 1$), L_i, R_i ($0 \le L_i \le R_i < N$), and X_i ($0 \le X_i \le 10\,000$ if $T_i = 0$, and $0 \le X_i < N$ if $T_i = 1$).

The last line contains the N integers S_p ($1 \le S_p \le 10\,000$), the number of seconds each director spends on answering a single email.

Output

Print a line containing the total time spent by the directors answering messages.

Examples

input	output
8 5 4 0 0 0 1 1 0 0 0 3 6 2 1 3 5 5 0 5 7 1 0 0 2 3 1 1 5 3 4 1 5 3 5 2 1 3	82

Explanation

Initially, the number of messages in the inboxes is $[4, 0, 0, 0, 1, 1, 0, 0]$.

On the first day directors $3, 4, 5$, and 6 receive 2 new messages each. The number of messages are $[4, 0, 0, 2, 3, 3, 2, 0]$ now.

On the second day,

- director 3 sends their messages to director $3 \oplus 5 = 6$,
- director 4 sends their messages to director $4 \oplus 5 = 1$,
- director 5 sends their messages to director $5 \oplus 5 = 0$.

The number of messages are $[7, 3, 0, 0, 0, 0, 4, 0]$ now.

On the third day directors $5, 6$, and 7 receive 1 new messages each. The number of messages are $[7, 3, 0, 0, 0, 1, 5, 1]$ now.

On the fourth day directors $0, 1$, and 2 receive 3 new messages each. The number of messages are $[10, 6, 3, 0, 0, 1, 5, 1]$ now.

On the last day,

- director 1 sends their messages to director $1 \oplus 3 = 2$,
- director 2 sends their messages to director $2 \oplus 3 = 1$,
- director 3 sends their messages to director $3 \oplus 3 = 0$,
- director 4 sends their messages to director $4 \oplus 3 = 7$,
- director 5 sends their messages to director $5 \oplus 3 = 6$.

The number of messages are $[10, 3, 6, 0, 0, 0, 6, 1]$ now.

Finally, the total amount of time they spend on answering messages is $10 \cdot 4 + 3 \cdot 1 + 6 \cdot 5 + 0 \cdot 3 + 0 \cdot 5 + 0 \cdot 2 + 6 \cdot 1 + 1 \cdot 3 = 82$ seconds.

Journey with Spoils

Kazakhstan is a country famous for many things, including the local, high-quality vodka. After competing at the ICPC World Finals in Astana, the Hungarian delegation decided to stock up and bring home as many bottles as possible. But how much is that exactly? Every country they visit on the way home has strict restrictions on the alcohol travelers can carry with them: it has a minimum age limit and a maximum amount per person. Luckily for the delegation, the plane tickets for the way home have not been purchased yet.

The delegation consists of K members numbered from 0 to $K-1$. The age of member i is represented by a positive integer A_i.

There are N airports numbered from 0 to $N-1$. Airport i requires the age of a passenger to be greater than or equal to B_i to enter when carrying vodka. Each person meeting the age limit is allowed to carry at most C_i liters of alcohol at this airport. There are M bidirectional flights connecting some pairs of airports. None of the flights start and end at the same location, and there is at most one flight between a pair of airports. It is possible to travel from any airport to any other airport by taking one or more flights.

The delegation starts at airport 0 and wants to travel to airport $N-1$ by taking one or more flights. The members can only enter an airport if they meet the restrictions of that airport. They are allowed to rearrange the vodka among themselves to maximize the total amount that they carry. Your task is to find the maximum amount of vodka that they can bring home.

Input

The first line of the input contains the integer K ($1 \leq K \leq 100\,000$), the number of delegation members. The second line contains K integers A_i ($0 \leq A_i \leq 10^9$), the age of each member.

The third line of the input contains the integers N ($2 \leq N \leq 100\,000$), the number of airports. Each of the next N lines contains two integers. The i-th line contains B_i and C_i ($0 \leq L_i \leq 10^9$, $0 \leq C_i \leq 10^9$), describing the restrictions of airport i.

The next line contains the integer M ($1 \leq M \leq 100\,000$), the number of available flights. Each of the next M lines contains two integers X_i and Y_i ($0 \leq X_i, Y_i < N$) representing two airports connected by a flight.

Output

Print a single line containing a positive integer: the maximum amount of vodka the delegation can take from airport 0 to airport $N-1$.

Examples

input	output
3 18 21 20 4 18 3 21 8 20 5 18 6 4 0 1 0 2 1 3 2 3	9

Explanation

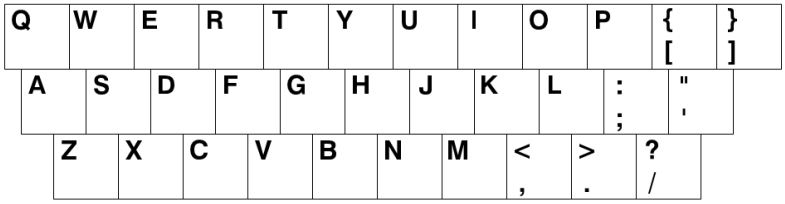
One possible way is to let all three members carry 3 liters of vodka each to airport 0, then take the flight to airport 2 and redistribute the vodka such that member 1 carries 5 liters and member 2 carries 4 liters. Finally, take the flight from airport 2 to airport 3 without rearrangement.

It can be proved that it is impossible to take more than 9 liters of vodka from airport 0 to airport 3.

Keyboard Mishap

The Kazakhstan Competitive Programming Federation has a very skilled blind secretary, Aisha. She is especially good at blind typing. Unfortunately, sometimes her hands are mispositioned on the keyboard and she consistently hits the right neighbor of every key or the left neighbor of every key. She is never off by more than one position to the left or the right, and she might as well be correctly positioned. She uses the standard US keyboard layout that is depicted in the picture below.

So, for example, if Aisha tries to type the word `fortune`, she might type `gptyimr` or `dierybw` or `fortune`. However, she notices if she hits a key that is bigger than a character key (e.g. shift or caps lock) and then corrects herself, so if she tries to type `salt`, it can only result in `ds;y` or `salt` because the left neighbor of the key `a` is caps lock, so she cannot be mispositioned to the left.



The keys on Aisha's keyboard.

You received a message from Aisha. You know that she *intended to write* a single word (not necessarily meaningful) that contains *only lowercase English characters*. Based on the string that you received, determine what could have been the word that Aisha tried to type. If there are multiple possible options, list all of them. It can also happen that there is no possible original word. (Maybe Aisha's cat was walking on her keyboard?)

Input

The first line contains a single string S without spaces, the message you received. The length of S is at most 100. It only contains characters that can be typed using the keys in the picture above, without pressing the shift key, i.e. the following characters: `qwertyuiop[]asdfghjkl;'zxcvbnm,./`

Output

In the first line of the output, print a single integer K, the number of possible original words. In the next K lines, print the possible original words, one per line, **in lexicographical order**. (If there are no possible original words, K is 0 and you don't have to print more lines.)

Examples

input	output
ytrr	3 tree uytt ytrr
adj[i][j].second	0
ailoveyou	1 ailoveyou

Explanation

In the first sample case, the received string is `ytrr`, and there are 3 possible original strings: `tree`, `uytt`, and `ytrr`. Note that they should be printed in lexicographical order.

In the second sample case, there isn't any possible original string because the `]` character key is not a neighbor of any letter key. Must have been the cat...

In the third sample case, the only option for the original string is the same as what we received.

Local Currency

The ~~Dictator~~ President of Kazakhstan is extremely proud that Astana was selected to host the ICPC World Finals. To celebrate this great occasion, he releases a special collection of tenge[1] coins. They released N kinds of new coins worth $2^0, 2^1, 2^2, \ldots, 2^{N-1}$ tenge, respectively.

When visiting Kazakhstan, you draw some cash from an ATM to buy souvenirs at the bazaar. The ATM gives you the full amount in these new coins: for each i between 0 and $N-1$ (inclusive), you receive A_i coins worth 2^i tenge.

You want to buy souvenirs from M costers, numbered from 0 to $M-1$. The costers are selling different items, and you quickly calculate that for each i, you need to pay B_i tenge to coster i to get all the items you want. The costers are crafty so if they receive more money for their items than their worth, they will refuse to give back any change! You find this incredibly annoying, and wonder whether you can pay **exactly** B_i tenge to coster i for every i from 0 to $M-1$ using your coins.

Input

The first line of the input contains two integers N ($1 \leq N \leq 50$) and M ($1 \leq M \leq 100\,000$), the number of coin types available to you, and the number of costers, respectively.

The second line contains N integers A_i ($0 \leq A_i \leq 10^{15}$), describing the number of coins you have from each kind. The third line contains M integers B_i ($0 \leq B_i \leq 10^{15}$), the amount of tenge you must pay each coster.

Output

Print a single line containing the string `YES` if it is possible to pay every coster the exact amount of tenge. Otherwise, print `NO`.

Examples

input	output
3 2 2 3 1 5 6	YES
3 2 1 2 2 5 3	NO

Explanation

In the first sample case, you can give one coin worth 1 and one coin worth 4 tenge to the first coster. Then, give three coins worth 2 tenge each to the second coster. At this point, you paid the required amount to both and left with one coin worth 1 tenge.

[1] the official currency of Kazakhstan