Task 1: `Voting Cities`

The great Emperor, *Lord Pooty*, decided to retire and would like to hand over the crown to one of his many sons. In the spirit of democracy, he decided to do this with a vote! His kingdom consists of N cities labelled from 0 to $N-1$. Of these N cities, K of them are voting cities where voting can be done. The ith voting city is T_i.

As a reponsible member of society, you decided that it is only right for you to do your civic duty. You are to travel to one of the designated voting cities to vote! There are E roads that can be used. Road j connects city U_j to city V_j **in one direction** and has a toll of C_j. Luckily, due to this event, local cities have opened a ticket system to reduce the cost of travelling.

There are 5 different types of tickets to choose from, numbered from type 1 to type 5. A ticket of type x will reduce the cost of the toll on a road by $(10x)\%$. In other words, the cost of the road will be multiplied by $\left(1-\frac{x}{10}\right)$ if a ticket of type x is used.

However, there are a few rules regarding the tickets. You cannot use more than one ticket on one road to stack the effects. You are only allowed to buy at most one of each ticket at the start of your journey. For example, you can choose to buy one type 1 ticket and one type 2 ticket but are not allowed to buy two type 2 tickets. This is to prevent people from hoarding the tickets. You are only allowed to buy the tickets at the start of your journey.

You are a busy man and unfortunately, you do not know which city you may start your journey from, nor do you know the ticket prices. You have made a list of Q possible situations, comprised of a starting city S and ticket prices P_1, P_2, P_3, P_4 and P_5 for the 5 tickets. **It is possible that a certain ticket may not even be available, and in that case the ticket price will be** -1.

For each of these situations, find the minimum cost to one of the voting city if it is reachable by road. Do note that not every city is reachable from every other and you may have to walk...

Input

Your program must read from standard input.

The first line of input contains 3 integers N, E and K representing the number of cities, number of roads and number of voting cities respectively. The second line contains K integers, the ith one representing T_i, the ith voting city.

The next E contain 3 integers each. The jth of these lines consists of U_j, V_j and C_j respectively, representing a unidirectional road from U_j to V_j with cost C_j. **It is guaranteed that C_j is divisible by 10**.

The next line contains a single integer Q, representing the number of situations to be considered.

The next Q lines contain 6 integers S, P_1, P_2, P_3, P_4 and P_5 representing the starting city and the prices of the tickets of type 1 to type 5 respectively. Note that the starting city and ticket prices can differ across the different situations provided.

Output

Your program must print to standard output.

Output Q lines with 1 integer on each line, representing the lowest cost to a voting city for each situation in the order provided in the input. If a path does not exist for a situation, print -1 instead.

Subtasks

The maximum execution time on each instance is 1.0s. For all testcases, the input will satisfy the following bounds:

- $1 \leq N \leq 5000$
- $0 \leq E \leq 10000$
- $1 \leq Q \leq 100$
- $0 \leq K \leq N$
- $0 \leq T_i < N$ for all $1 \leq i \leq K$
- $T_i \neq T_j$ for all $1 \leq i < j \leq K$
- $1 \leq C_i \leq 10^9$ for all $1 \leq i \leq E$
- C_i is a multiple of 10 for all $1 \leq i \leq E$
- $0 \leq U_i, V_i < N$ and $U_i \neq V_i$ for all $1 \leq i \leq E$
- $-1 \leq P_i \leq 10^9$ for all $1 \leq i \leq 5$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	5	There are no tickets sold. $P_i = -1$ for all i. $Q = 1$ and $K = 1$
2	5	There are no tickets sold. $P_i = -1$ for all i. $K = 1$
3	5	There are no tickets sold. $P_i = -1$ for all i.
4	5	$Q = 1$ and $K = 1$
5	5	$K = 1$
6	10	Each situation has at most 1 ticket available.
7	15	$1 \leq N \leq 100$, $1 \leq E \leq 1000$
8	50	-

Sample Testcase 1

This testcase is valid for subtasks 4,5, 7 and 8.

Input	Output

`3 2 1`
`2`
`0 1 100`
`1 2 200`
`1`
`0 10 20 1000 2000 -1`
	`280`

Sample Testcase 1 Explanation

In the only situation given, the optimal solution would be to buy the type 2 ticket and use it on the $1 \to 2$ road and the type 1 ticket and use it on the $0 \to 1$ road. This would have a cost of $200(1 - \frac{2}{10}) + 100(1 - \frac{1}{10}) + 10 + 20 = 160 + 90 + 10 + 20 = 280$.

Sample Testcase 2

This testcase is valid for all subtasks.

Input	Output

`2 0 1`
`1`
`1`
`0 -1 -1 -1 -1 -1`
	`-1`

Sample Testcase 2 Explanation

There is no road from your starting point to the only voting city. Thus, output -1.

Sample Testcase 3

This testcase is valid for subtasks 7 and 8.

Input	Output

`6 3 2`
`4 5`
`0 4 100`
`1 4 200`
`2 5 300`
`4`
`0 -1 -1 -1 -1 -1`
`1 20 40 10 100 4`
`2 1 2 3 4 0`
`3 0 -1 0 0 0`
	`100`
	`104`
	`150`
	`-1`

Task 2: `Gym Badges`

In a quest to be the very best, you have decided to set out on your journey around the region to prove your strength by collecting gym badges. You begin your solo adventure with your first and only Pokemon, which happens to be a legendary *Wabbit*.

Your Wabbit starts at level 0 and **can only** gain levels by challenging gyms. There are N gyms numbered from 1 to N spread across the region and you can challenge them in any order. To prevent over grinding, gym i has its own level cap L_i where you can only challenge the gym if Wabbit's current level is **less than or equal to** L_i.

As there may be different number of trainers to defeat in a gym, the number of levels that Wabbit gains after challenging a gym may differ. To be precise, Wabbit will gain X_i levels after challenging gym i.

Each gym i rewards successful challengers with their own unique gym badge i. Find the maximum number of unique gym badges that you can obtain if you challenge the gyms in an optimal way.

Input

Your program must read from standard input.

The first line contains an integer N, the number of gyms.

The second line contains N integers, where the i^{th} integer represents the level Wabbit gains by challenging i^{th} gym, X_i.

The third line contains N integers, where the i^{th} integer represents the level cap of the i^{th} gym, L_i.

Output

Your program must print to standard output.

The output should contain a single integer on a single line, the maximum number of unique gyms badges that can be won.

Subtasks

The maximum execution time on each instance is 1.5s. For all testcases, the input will satisfy the following bounds:

- $1 \leq N \leq 500\,000$
- $1 \leq X_i, L_i \leq 10^9$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	15	$1 \leq N \leq 10$
2	9	L_i is constant
3	27	$1 \leq N \leq 5000$
4	49	-

Sample Testcase 1

This testcase is valid for subtasks 1, 2 and 4.

Input	Output

`5`
`4 6 3 5 2`
`10 6 4 8 12`
	`4`

Sample Testcase 1 Explanation

An optimal solution can be obatined when the gyms are challenged in the following order: $3, 4, 1, 5$.

Sample Testcase 2

This testcase is valid for all subtasks.

Input	Output

`5`
`3 9 4 2 6`
`10 10 10 10 10`
	`4`

Sample Testcase 2 Explanation

An optimal solution can be obatined when the gyms are challenged in the following order: $1, 3, 4, 2$.

Note that after the challenging gym 4, Wabbit is at level 9 which is within the level cap of gym 2 and can thus challenge it.

At the end Wabbit will be at level 18 which is above the level cap of the gym 5 and hence cannot challenge it.

Task 3: `Towers`

Benson the Rabbit likes towers. There are N cities numbered 1 to N, and city i is located at a point with integer coordinates (X_i, Y_i). No two cities are located at the same point. Benson wants to build towers in some of these cities such that the following conditions are satisfied:

- For any a, there are at most two towers with x-coordinate equal to a.
- For any b, there are at most two towers with y-coordinate equal to b.
- Each of the N cities either has a tower built, or lies on the line segment between two towers with the same x-coordinate or the same y-coordinate. More formally, for a city located at (x, y), if there is no tower in that city, then there are two towers at coordinates $(x, c), (x, d)$ with $c \leq y \leq d$, or two towers at coordinates $(e, y), (f, y)$ with $e \leq x \leq f$.

Benson knows that it is always possible to build towers satisfying these conditions, but does not know how he should do so. Help Benson determine where he should build the towers.

Input

Your program must read from standard input.

The first line of the input contains one integer, N, the number of cities.

In the next N lines, the i^{th} line contains two integers X_i, Y_i, which means city i is located at the point (X_i, Y_i).

Output

Your program must print to standard output.

Output one line containing a string of N characters $A_1 A_2 \ldots A_N$. A_i should be 1 if Benson should build a tower in city i, and 0 otherwise. The towers built should satisfy all the conditions.

If there are several solutions your program can output any one of them.

Subtasks

The maximum execution time on each instance is 2.0s. For all test-cases, the input will satisfy the following bounds:

- $1 \leq N \leq 10^6$
- $1 \leq X_i, Y_i \leq 10^6$
- For all $i \neq j$, either $X_i \neq X_j$ or $Y_i \neq Y_j$.

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	5	$N \leq 3$
2	11	$N \leq 16$
3	7	$N = ab$ for some positive integers a, b and $(X_{ai+j}, Y_{ai+j}) = (i+1, j)$ for all integers i, j with $0 \leq i \leq b-1, 1 \leq j \leq a$
4	6	For every integer a, there are at most two cities whose x-coordinate is a.
5	31	$N \leq 5000$
6	21	$N \leq 100000$
7	19	-

Sample Testcase 1

This testcase is valid for subtasks 1, 2, 5, 6, and 7.

Input	Output

`3`
`1 1`
`1 6`
`1 5`
	`110`

Sample Testcase 1 Explanation

If we build towers in cities 1 and 2 which have the same x-coordinate, city 3 which also has the same x-coordinate lies on the line segment between them.

An example of an incorrect output would be 111, as there will be more than two towers having x-coordinate equal to 1 if towers are built in all 3 cities.

Another incorrect output is 101, as even though city 2 shares the same y-coordinate with cities 1 and 3, it does not lie on the line segment between them.

Sample Testcase 2

This testcase is valid for subtasks 2, 3, 4, 5, 6, and 7.

Input	Output
`6`	`110011`
`1 1`	
`1 2`	
`2 1`	
`2 2`	
`3 1`	
`3 2`	

Sample Testcase 2 Explanation

City 3 lies on the line segment between cities 1 and 5 which have the same y-coordinate, and city 4 lies on the line segment between cities 2 and 6 which have the same y-coordinate.

Sample Testcase 3

This testcase is valid for subtasks 2, 5, 6, and 7.

Input	Output

`8`
`1 13`
`2 13`
`7 27`
`7 13`
`7 2`
`2 27`
`7 4`
`4 13`
	`10101101`

Task 4: `Grapevine`

Syrup the Turtle waters the huge Grapevine which snakes around town. The layout of the Grapevine can be best described as having N leafy joints, which Syrup has numbered 1 to N, connected by $N - 1$ branches also numbered 1 to $N - 1$. Each branch i directly connects two joints A_i and B_i, and has a length of W_i. No two branches directly connect the same pair of joints, and as part of the single Grapevine, every joint is connected to every other either directly or indirectly by branches.

With his sturdy hose and a little handiwork, Syrup is able to sway the growth of the Grapevine as he deems fit. In particular, he can **soak** a joint of the vine - causing it to immediately sprout a single giant grape if it was empty, or instead dislodging the grape there if one was present.

He can also **anneal** a branch with water, extending or shortening it to a new length w_i by spraying at the right pressure and angle. To make sure things are on track, Syrup will periodically stand atop an elevated joint and **seek** for the closest grape. The distance from such a joint to a grape is defined by the shortest path starting from said joint, traversing a number of branches (or none at all), and ending at the grape's own joint.

Right now, the Grapevine has no grapes attached after the last passing storm. Syrup has his watering agenda of Q actions planned out for the week and is about to begin spraying; but first, he needs to know what distances to expect when he **seeks** for grapes along the way. Given Syrup's watering plans, your task is to find for each planned **seek** the distance from the specified joint to the nearest grape.

Input

Your program must read from standard input.

The first line contains two integers, N and Q.

$N - 1$ lines will follow. The i^{th} line contains three integers, A_i, B_i, and W_i, describing a single branch.

Q lines will follow, each representing an action by Syrup.

- If the first integer of the line is 1, it represents a **seek** and 1 integer q_i will follow. This means that you are to determine the distance between joint q_i and the nearest joint with a grape, and output this distance. If there are no grapes on the Grapevine, output -1 instead.
- If the first integer of the line is 2, it represents a **soak** and 1 integer u_i will follow. This

means that joint u_i is soaked and grows a grape or has its grape dislodged.

- If the first integer of the line is 3, it represents an **anneal** and 3 integers a_i, b_i, and w_i will follow. This means that the length of the branch between joints a_i and b_i has had its length changed to w_i. It is guaranteed that a branch between joints a_i and b_i exists.

Output

Your program must print to standard output.

For each **seek** action, output one line with a single integer, the distance to the closest grape, or -1 if no grapes are present.

Subtasks

The maximum execution time on each instance is 3.0s. For all testcases, the input will satisfy the following bounds:

- $2 \leq N \leq 100\,000$
- $1 \leq Q \leq 100\,000$
- $1 \leq A_i \neq B_i \leq N$
- $0 \leq W_i \leq 10^9$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Points	Additional Constraints
1	6	$N, Q \leq 2000$
2	14	For all **seek** actions, $q_i = 1$
3	15	The vine forms a complete binary tree, $A_i = \lfloor \frac{i+1}{2} \rfloor$, $B_i = i + 1$
4	15	There is at most 1 grape on the vine at any point in time.
5	18	All **soak** actions will occur before any **seek** or **anneal** actions. For all **anneal** actions, $w_i = 0$
6	32	-

Sample Testcase 1

This testcase is valid for subtasks 1, 2, 5, and 6.

Input	Output
`7 11`	`11`
`1 2 2`	`8`
`2 3 4`	`8`
`5 6 1`	`2`
`5 3 6`	
`3 7 6`	
`2 4 9`	
`2 6`	
`2 4`	
`2 7`	
`1 1`	
`3 2 3 0`	
`1 1`	
`3 6 5 0`	
`1 1`	
`3 3 5 0`	
`3 2 4 0`	
`1 1`	

Sample Testcase 1 Explanation

In the first **seek**, the closest grape is at joint 4 over the path $1 \longrightarrow 2 \longrightarrow 4$, for a distance of $2 + 9 = 11$. The second **seek** has the closest grape at joint 7, while the third **seek** has both the grapes at joints 6 and 7 closest. The fourth and final **seek** has its closest grape at joint 4.

Sample Testcase 2

This testcase is valid for subtasks 1, 3, and 6.

Input	Output
6 11	7
1 2 3	2
1 3 4	-1
2 4 1	4
2 5 4	
3 6 6	
2 3	
1 2	
2 4	
3 1 3 2	
1 1	
2 3	
3 2 1 2	
2 4	
1 3	
2 2	
1 3	

Sample Testcase 3

This testcase is valid for subtasks 1, 4, and 6.

Input	Output
7 8	3
1 2 2	11
2 3 3	0
3 6 2	8
4 6 1	
5 6 4	
6 7 3	
2 3	
1 4	
2 3	
2 5	
1 1	
3 6 7 4	
1 5	
1 7	

Task 5: `Fruits`

Supermarkets usually have fruits for sale in sections, where each section is dedicated to a single type of fruit. The supermarket that Benson the Rabbit is visiting has N sections and N types of fruits. The sections are numbered from 1 to N and each fruit is numbered from 1 to N.

The ith fruit has tastiness i and cost C_i. **It is guaranteed that $C_i \le C_j$ for all $1 \le i < j \le N$.**

Each of the N sections is to be assigned a **distinct** type of fruit. At the moment, the type of fruit assigned to section j is A_j. If $A_j = -1$, then section j is empty. Otherwise, fruit A_j is already assigned to section j. Once all N fruits have been assigned, the supermarket will open and Benson will enter the supermarket to buy the fruits.

Benson is very picky but also in a rush, so he will visit the sections in increasing order. Benson's basket is initially empty, and when he reaches a section, he will compare the tastiness of the fruit in that section to the tastiness of all of the fruits in his basket. If his basket is empty, or if the tastiness of the fruit at the current section is **greater than the tastiness of every other fruit in his basket**, Benson will add that fruit to his basket.

To maximise revenue, you have been tasked with assigning the fruits to the sections such that the sum of cost of fruits that Benson adds to his basket is maximised. As Benson is rushing for time, he might only visit the first few sections before going straight to the cashier. Help compute, for each k from 1 to N, the maximum possible revenue that can be achieved if Benson only visits the first k sections **given that the arrangement of fruits can change for different k.**

Input

Your program must read from standard input.

The input starts with a line with one positive integer N.

The second line contains N integers where the ith integer represents A_i.

The third line contains N integers where the ith integer represents C_i.

Output

Your program must print to standard output.

The output should contain a N integers on a single line. The kth one should be the maximum sum of costs that Benson will pay if the fruits are assigned optimially if he visits only the first

k sections.

Subtasks

The maximum execution time on each instance is 1.0s. For all testcases, the input will satisfy the following bounds:

- $1 \le N \le 400\,000$
- $1 \le A_j \le N$ or $A_j = -1$
- $1 \le C_i \le 10^9$
- $C_i \le C_j$ for all $1 \le i < j \le N$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	6	$N \le 8$
2	5	$A_j = -1$ for all $0 \le j < n$
3	11	$N \le 200$
4	13	$N \le 2000$
5	23	$C_i = 1$ for all $0 \le i < n$
6	42	-

Sample Testcase 1

This testcase is valid for all subtasks.

Input	Output

5
-1 -1 -1 -1 -1
1 1 1 1 1
	1 2 3 4 5

Sample Testcase 1 Explanation

We can arrange the fruits in increasing order (1,2,3,4,5). Benson will take all fruits that he passes by, so for each k from 1 to 5, Benson will take k fruits which have a total cost of k.

Sample Testcase 2

This testcase is valid for subtasks 1, 3, 4 and 6.

Input	Output
`5`	`3 4 7 9 9`
`-1 3 -1 -1 -1`	
`1 2 2 2 3`	

Sample Testcase 2 Explanation

If Benson only visits the first section, it is optimal to put fruit 5 in section 1. This gives a cost of 3.

If Benson only visits the first 2 sections, it is optimal to put fruit 2 in section 1 and fruit 3 in section 2. This gives a cost of $2 + 2 = 4$.

If Benson only visits the first 3 sections, it is optimal to put fruit 2 in section 1 and fruit 3 in section 2 and fruit 5 in section 3. This gives a cost of $2 + 2 + 3 = 7$.

If Benson visits either the first 4 or all 5 sections, it is optimal to put the fruits in the order $2, 3, 4, 5, 1$. This gives a cost of $2 + 2 + 2 + 3 = 9$.

Sample Testcase 3

This testcase is valid for subtasks $3, 4, 5$ and 6. Note that the second line of input has been split into 2 lines for clarity's sake.

Input	Output
`13`	`1 2 3 4 5 6 6 7 8 9 9 9 9`
`-1 -1 5 6 -1 -1 7 11 -1 -1 10`	
`-1 -1`	
`1 1 1 1 1 1 1 1 1 1 1 1 1`	

Sample Testcase 4

This testcase is valid for subtasks 3, 4 and 6. Note that the first line of output has been split into 2 lines for clarity's sake.

Input	Output

`10`
`-1 -1 -1 -1 5 -1 -1 -1 9 -1`
`5 11 24 27 35 60 72 81 91 92`
	`92 173 245 305 305 332 356 367`
	`406 498`