Problem A. Arrangement

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Colin and his friends were invited to set problems for the 5th Guangxi Collegiate Programming Contest!

They were so excited that everyday they gathered together to come up with some awesome ideas. Within several days, they come up with n problems, marked from A (followed by B, then C ext.). Aiming at bringing contestants a good experience, they need to measure the difficulty of the problems. They put forward the following two indices for each problem:

- **Idea Difficulty**: the difficulty of coming up the solution for this problem
- **Coding Difficulty**: the difficulty of programming to solve the problem

Then, Colin defines the **Final Difficulty** of a problem as **Idea Difficulty times Coding Difficulty** (i.e. Final Difficulty = Idea Difficulty $\times$ Coding Difficulty).

Now, Colin tells Eva the **Idea Difficulty** and **Coding Difficulty** of each problem, and asks her to arrange the problems in ascending order according to the following rules:

- If the Final Difficulty of two problems are different, the problem with lower Final Difficulty ranks first
- otherwise, the problem with lower Idea Difficulty ranks first
- otherwise, the problem with smaller mark ($A < B < C < \dots$) ranks first

However, Eva don't know how to solve this task, can you help her to arrange the problems in ascending order?

Input

The first line is a number n ($n \leq 26$) , representing the number of problems.

The second line consists of n numbers $a_1, a_2, \cdots, a_n$ ($1 \leq a_i \leq 10^5$), representing the Idea Difficulty of n problems (that is, the first number is the Idea Difficulty of problem A, the second one is the Idea Difficulty of problem B...and so on)

The third line consists of n numbers $b_1, b_2, \cdots, b_n$ ($1 \leq b_i \leq 10^5$), representing the Coding Difficulty of n problems (that is, the first number is the Coding Difficulty of problem A, the second is the Coding Difficulty of problem B ... and so on)

Output

A line of n uppercase letters, separated by a space, representing the ascending order of n problems.

Example

standard input	standard output
3 1 2 3 3 2 1	A C B

Problem B. Bus

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Gui Lin Shan Shui Jia Tian Xia!

Hearing of the scenery of Guilin for a long time, Colin, as the monitor, decided to take his classmates to Guilin to have a good time.

Considering the cost and the pleasure of the journey, they chose bus as the transportation.

There're n students in Colin's class(including him) and each bus can hold m students.

Could you help Colin to tell the travel agent how many buses they need?

Input

Two integer, $n, m (1 \leq n, m \leq 4 \times 10^{15})$, representing the number of Colin's classmates and the capacity of the bus.

Output

If they only need one bus, output "We need one bus.". Otherwise, if they need k buses, output "We need k buses."(without quotes).

Examples

standard input	standard output
10 5	We need 2 buses.
7 20	We need one bus.

Problem C. Counting Strings

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Eva heard that Colin recently learned string algorithm. In order to stumped Colin, Eva came up with a string problem.

Not surprisingly, Colin was stumped by this question. As a good friend of Colin, it is necessary for you to step up and help Colin solve the problem.

Now you have n strings consisting of **lowercase** letters in your hand.

And then Eva will ask you q questions. In each question , Eva will give you two strings a , b. You should answer that how many strings s_i in your hand satisfies that "a is its **prefix** and b is its **suffix**".

A **prefix** of a string S is any leading contiguous part of S.

A **suffix** of a string S is any trailing contiguous part of S.

For example, "c"and "colin"are prefixes , and "a"and "eva"are suffixes of the string **"colinloveseva"**.

Input

The first line of each test case contains two integers n , q ($1 \leq n, q \leq 10^5$) — the number of strings and the number of queries.

The next n lines; each line contains one **non-empty** string s_i — the i-th string.

The next q lines; each line contains two **non-empty** string a_i , b_i — the i-th prefix and suffix of the query.

We guarantee that the testdata meets the following conditions:

$$\sum_{i=1}^{n} \text{length } s_i \leq 10^6$$

$$\sum_{i=1}^{q} \text{length } a_i + \text{ length } b_i \leq 10^6$$

Output

For each test case, print q lines, each line contains an integer.

Example

standard input	standard output
5 5 bbaabab bbbbbab aaba abbab bbab a aa a ab bb aa ba aa b b	0 1 0 0 3

Problem D. Driving

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Life moves pretty fast. If you don't stop and look around once in a while, you could miss it.

Successfully passing the final exam, Colin and Eva decided to take a road trip in Guangxi province to enjoy their youth.

They plan to drive themselves so they can enjoy the scenery along the way.

The map consists of n cities and m one-way roads. They will depart from City S to City T with the initial speed $v = 1$. They can safely pass a d-meter road in $\lceil \frac{d}{v} \rceil$ minutes. ($\lceil n \rceil$ means the smallest integer which aren't less than n) Colin can't wait for the view of City T and wishes he could get there faster. Some auto-repair centers are ready to serve him.

There are p auto-repair centers. The i_{th}-auto-repair center can double the speed of their car by a c_i-minute upgrade. (Of course you can stay in this center to upgrade your car multiple times, but every upgrade will take c_i minutes indipendently.)

However, some of the roads are of very poor quality. Colin and Eva can drive across them spending the same time as normal road, but the speed of their car will go back to 1 at the end of road.

Colin wants to know how long it will take him to reach City T at least.

Could you help him?

Input

The first line contains four integer $n, m, S, T (1 \le n \le 2 \times 10^4, 1 \le m \le 7 \times 10^4, 1 \le S, T \le n)$.

Each line of the following m lines contains three integers $a, b, c (1 \le a, b \le n, 1 \le c \le 10^6)$ and a character ch which equals 'G' or 'B', representing a one-way road begins at a and ends at b with c meters long. If ch equals 'G' means this road is of good quality, while 'B' means the road is of poor quality.

Then the following line consists of a single integer $p (1 \le p \le n)$.

The following p lines, each line consists of two integers $x_i, c_i (1 \le x_i \le n, 1 \le c_i \le 10^6)$, representing the auto-repair center at City x_i, needs c_i minutes to double the speed of Colin's car.

Output

An integer, representing the minimal time they need to reach City T.

If they can't arrive in City T, print '-1'.

Examples

standard input	standard output
5 4 1 5 1 2 4 G 2 3 1 G 3 4 9 G 4 5 10 G 1 1 1	8
5 4 1 5 1 2 4 G 2 3 1 B 3 4 9 B 4 5 10 G 1 1 1	23

Problem E. Envy-freeness

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Fair division of goods among competing agents is a fundamental problem in Economics and Computer Science. There is a set M of m goods and the goal is to allocate goods among n agents in a fair way. When can an allocation be considered "fair"? One of the most well-studied notions of fairness is Envy-freeness.

An allocation is a partition of M into disjoint subsets $X_1, \ldots, X_n$ where X_i is the set of goods given to agent i . Every agent i has a value associated with each good j ,which represented as $w_{i,j}$. And every agent values a set of goods S as the sum of the values of the goods in S, which represented as $W_i(S) = \sum_{j \in S} w_{i,j}$.

Then we define that agent i **envies** agent j if i values X_j more than X_i , i.e. $W_i(X_j) > W_i(X_i)$.

An allocation is **"envy-free"**(represented as **EF**) if no agent envies another, however, it's not always exists.

Then we define **"envy-free up to any good"**(represented as **EFX**) : In an EFX allocation, agent i may envy agent j , however this envy would vanish as soon as **any** good is removed from X_j . i.e. if $W_i(X_j) > W_i(X_i)$, then $\forall g \in X_j, W_i(X_j \setminus g) \leq W_i(X_i)$. However, an EFX allocation may not exist too.

Then we define **"envy-free up to one good"**(represented as **EF1**) : In an EF1 allocation, agent i may envy agent j , however this envy would vanish as soon as **a** good is removed from X_j . i.e. if $W_i(X_j) > W_i(X_i)$, then $\exists g \in X_j, W_i(X_j \setminus g) \leq W_i(X_i)$.

Note that in EFX and EF1, no good is really removed.

Today Colin and Eva wants to study this problem. To simplify the problem, we considered there only exists two agents: Colin (agent 1) and Eva (agent 2). At first neither of them had any goods. Then there comes m operations (allocate m goods) , each operation will provide three values $c_i, e_i, b_i = 0/1$ to represent one good , which means the value of this good in **Colin's perspective** (i.e. $W_{1,i} = c_i$) ,the value of this good in **Eva's perspective** (i.e. $W_{2,i} = e_i$) , and who it was allocated to (i.e. if $b_i = 0$, then it was allocated to Colin, otherwise it was allocated to Eva)

They want you to tell them, after each operation, what is the highest situation for the current allocation ?

We define the priority as: **"envy-free"** is the highest, **"envy-free up to any good"** is the second, **"envy-free up to one good"** is the third, and if none of the three conditions are satisfied, the worst is "**envy**".

Input

The first line contains one integer m ($1 \leq m \leq 10^6$)

Each of the following m lines contains three integers c_i, e_i ($1 \leq c_i, e_i \leq 10^6$) and b_i ($b_i \in \{0, 1\}$)

Output

After each operation, output a single line with:

- If the current allocation is **"envy-free"**, then output "EF"
- Otherwise if the current allocation is **"envy-free up to any good"**, then output "EFX"
- Otherwise if the current allocation is **"envy-free up to one good"**, then output "EF1"
- Otherwise output "E"

Example

standard input	standard output
5 5 2 0 5 2 1 2 2 0 9 2 1 9 2 1	EFX EF EFX EF1 E

Problem F. Finding Stars

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Eva's birthday is coming soon, Colin is picking out a birthday present for eva. "There is no gift more romantic than a star."Colin thought. So Colin decided to pick out some stars from the night sky. But Colin has some special requirements for the stars. As Colin's good friend, it's time to help him.

There are now **1000 kinds** of stars in the sky.

And Colin has an array A of N stars . A_i describes that the i th star 's type is A_i. You have to answer Q queries of three types:

0. p x - change $A_p \rightarrow$ x

1. l r - Query whether the parity of all 1000 stars' number of occurence in the interval $[l, r]$ is the same (Output 1 only if all stars appear an odd number of times or all stars appear an even number of times, otherwise output 0)

2. l r - Query how many of all 1000 stars appear odd number of times in $[l, r]$

Input

The first line of each test case contains two integers n , q ($1 \leq n, q \leq 10^5$) — the number of stars and the number of querys.

The next line contains n integer A_i ($1 \leq A_i \leq 1000$) — the i-th star's type.

The next q lines contains three integer

$$\{op_i = 0, p_i, x_i\} \text{ or } \{op_i = 1, l_i, r_i\} \text{ or } \{op_i = 2, l_i, r_i\}$$

Guarantee that:

$$\forall op_i = 0,\ 1 \leq p_i \leq n, 1 \leq x_i \leq 1000$$

$$\forall op_i = 1, 1 \leq l_i \leq r_i \leq n$$

$$\forall op_i = 2, 1 \leq l_i \leq r_i \leq n$$

Output

For each $op_i = 1$ or 2 , print one line contains one integer.

Example

standard input	standard output
5 10 1 3 5 1 2 0 4 4 0 1 2 0 1 2 1 2 5 2 4 4 1 2 2 2 3 4 1 1 2 0 5 2 0 4 2	0 1 0 2 0

Problem G. Gambling

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	512 megabytes

Colin is going to take the oral English test recently, but Colin is not good at English.

The rules of the English test are as follows:

There are n questions in total, and the English teacher Eva will randomly draw one question with **equal probability**. Colin only needs to answer the question drawn by Eva, and his oral test score is the score he gets on this question.

Colin only prepared one of the questions. For unprepared questions, Colin can only get **50** points. But for prepared questions, Colin can get **100** points. Because Teacher Eva knew that Colin's English was not good, she gave Colin a chance to re-draw the question.

When Colin chose to re-draw a question , the question drawn for the first time will be thrown into the trash , Colin will not get points for it. And then , the score obtained this time will be deducted 20% (that is, **only** 80% **can be obtained of the score**). **Notice that**: When re-drawing, teacher Eva will also randomly draw a question from the remaining $n-1$ questions with **equal probability**.

Colin is smart, and he chooses the decisions that work in his favor. What is the **maximum expected score** Colin can get under the optimal decision ?

For ease of calculation, you only need to output **the result of multiplying the answer by n.**

Input

One integer n ($2 \leq n \leq 10^9$) , representing the number of questions.

Output

One integer E, representing **maximum expected score** times n. It is guaranteed that the answer is an integer.

Example

standard input	standard output
2	180

Problem H. Homework

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	512 megabytes

Colin has n optional courses numbered from 1 to n, so he has to do a lot of homework.

For the i-th course, Colin must write an essay with no less than a_i words. And Colin can write one word per second.

However, Colin wants to finish homework as fast as possible, so he decides to reuse his homework. For the i-th course homework, he can write a_i words directly, or spend $c_i (c_i < a_i)$ seconds to copy and modify **previous finished** homework $j (j \neq i)$. If $a_j \geq a_i$, he will just **keep** a_i words. But if $a_j < a_i$, he has to write another $a_i - a_j$ words. Notice that Colin can finish his homework in any order.

Colin doesn't have enough time to make his homework plan because of the deadline, so he wants you to write a program to determine the order of doing homework so that he can finish homework as fast as possible(no matter write or copy).

Input

The first line contains one integer n $(1 \leq n \leq 10^5)$.

The second line contains n integers $a_1, a_2, \cdots, a_n$ $(2 \leq a_i \leq 10^5)$.

The third line contains n integers $c_1, c_2, \cdots, c_n$ $(1 \leq c_i < a_i)$.

Output

Print one integer representing the minimum number of seconds required to finish all homework.

Example

standard input	standard output
5 800 1500 1000 2000 3000 50 40 60 80 100	3230

Note

Finish the fifth homework first, and copy the fifth homework to others.

The number of seconds is $3000 + 50 + 40 + 60 + 80 = 3230$.

Problem I. Infinity

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

The number of stars is finite, while natural numbers are infinite. Count all the stars, and you will still have as many natural numbers left over as you started with.

Colin is poor at math. To help him pass the final exam of linear algebra, Eva prepared an arithmetical practice for him.

Eva will give Colin three integers a, b, n, and a variant x in the beginning.

Before Colin start the practice, he can choose zero or one as the initial value of x.

Then Colin will have two operations to choose:

1. Mulitply: let $x \leftarrow x \times a$.
2. Add: let $x \leftarrow x + b$.

Eva will ask him to progress x to n in finite steps by these two operations. For example, when $a = 5, b = 3, n = 40$, there's a legal progess:

- step 1: $x \leftarrow x \times a$, x equals to 5.
- step 2: $x \leftarrow x + b$, x equals to 8.
- step 3: $x \leftarrow x \times a$, x equals to 40.

But not every positive integer is possible to be generated in this way. When $a = 2, b = 4$, there's no way to change x into 3.

For a pair (a, b), Eva wants to know if there exists a positive integer N, satisfying that every positive integer n greater than N is able to be generated, so that she can easily create infinite questions for Colin.

Literally, define $n \Leftarrow (a, b)$ means n can be generated with (a, b). Then giving a pair (a, b), Eva wants to know if

$$\exists N, \forall n > N, n \Leftarrow (a, b)$$

Could you help her?

Input

Two integers a and b $(1 \leq a, b \leq 10^6)$.

Output

If there exists an integer N satisfying the requirement, output 'YES'(without qoutes). Otherwise, output 'NO'(without qoutes)

Example

standard input	standard output
3 5	YES

Problem J. Jump game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Colin and Eva are playing a jump game.

In this game, players are given n integers numbered from 1 to n $(a_1, a_2, \cdots, a_n)$. Players start at point 1, and must jump to point n to finish this game.

However, this game has a strange rule. Only when $i < j$ and $\gcd(a_i, a_j) > 1$ that players can jump from point i to point j. In addition, all n integers have a special limitation: a_i has **no more than** 5 prime factors.

Eva is so smart that she immediately comes up with a solution. Colin doesn't want to lose to Eva, so he asks you for help. He wants you to count the number of different ways to finish this game (Two ways are different if and only if the point on the paths of two ways are different).

The path of a legal way is $k_1, k_2, \cdots, k_m$ that $k_1 = 1$, $k_m = n$, $k_i < k_{i+1}$, $\gcd(a_{k_i}, a_{k_{i+1}}) > 1$.

Since the answer can be very large, you only need to print the answer mod $10^9 + 7$.

Recall that $\gcd(a, b)$ represents the greatest common divisor of a and b.

Input

The first line contains one integer n $(2 \le n \le 10^5)$.

The second line contains n integers $a_1, a_2, \cdots, a_n$ $(1 \le a_i \le 10^5)$.

For $1 \le i \le n$, a_i has **no more than** 5 prime factors.

Output

Print one integer representing the number of different ways $(\text{mod } 10^9 + 7)$ to finish this game.

Examples

standard input	standard output
5 2 42 15 35 5	3
3 1 2310 23333	0

Problem K. Kirby's Challenge

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

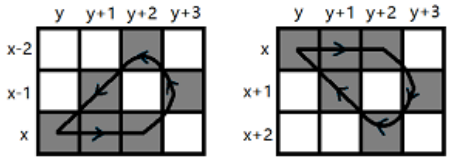
Recently, Colin bought a Switch for Eva. And they are playing "Kirby and the Forgotten Land".

In a challenge mission, Kirby is in a $4 \times n$ grid. The row of it is numbered from 1 to 4, and the column of it is numbered from 1 to n. There are many keys in this grid. Let $a_{x,y}$ represent the status of cell (x, y). If $a_{x,y} = 1$, there is a key in (x, y). If $a_{x,y} = 0$, there is no key in (x, y).

Kirby starts at $(1, 1)$, and should reach $(4, n)$. Moreover, Kirby must collect all the keys in the grid to open the door in $(4, n)$. Kirby will collect the key at (x, y) when Kirby reach (x, y). Of course, Kirby will collect the key at $(1, 1)$ at the beginning.

In a second, Kirby can move from (x, y) to $(x+1, y), (x, y+1), (x-1, y)$. Or Kirby can stay at (x, y) and throw a returnable flying weapon(boomerang) to collect keys in the flying path.

Kirby has two ways to throw the weapon. As the picture shows:



The flying path is represented as the grey cells, so keys in the grey cells can be collected by the weapon.

In a second, Kirby can only choose one way to throw the weapon, but Kirby can throw the weapon **multiple times** at (x, y) if necessary.

Notice: Kirby can't get off the grid, but the weapon can fly outside the grid and keep the flying path.

Please write a program to help Colin and Eva find the shortest time to complete the challenge mission, so that they can get more rewards.

Input

The first line contains one integer n $(1 \le n \le 100)$.

In the next 4 lines, the x-th line contains n integers $a_{x,1}, a_{x,2}, \cdots, a_{x,n}$ $(0 \le a_{x,y} \le 1)$.

Output

Print one integer representing the minimum number of seconds required to complete the challenge mission.

Example

standard input	standard output
5 1 1 1 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 1	8

Note

The best solution is: Spend 1 second to throw the weapon in the second way at $(1, 1)$, and spend 7 seconds to reach $(4, 5)$.

Problem L. Lowbit

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

One day Colin learned how to represent a number in binary form. For example, the binary form of $(11)_{10}$ is $(1011)_2$, and he was very excited!

Further, Eva taught Colin a well-known binary function lowbit(x) which represents the value of the bit corresponding to the lowest 1 of x in binary form. For example, lowbit(11) = 1 , lowbit(4) = 4

Colin thought this function was amazing, so he thought of an interesting problem, and then it comes:

Given a binary form of a number $x(x \leq 2^{10^5})$, you need to perform some operations on x.

Each turn you can choose one of the following two operations:

- $\mathbf{x+ = lowbit(x)}$, which means adding the lowest bit of x to x
- $\mathbf{x- = lowbit(x)}$, which means subtracting the lowest bit of x from x

Now Colin wants to know, what is the minimum number of operations required to turn x into 0?

Input

A single line with the binary form of x $(1 \leq x \leq 2^{10^5})$ without leading zero.

Output

One integer, the minimum number of operations required.

Example

standard input	standard output
1100101	4