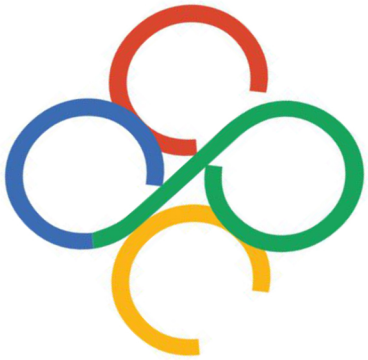
2022 Hubei Provincial Collegiate Programming Contest

2022-05-22

武漢大學
WUHAN UNIVERSITY



Problems

A. Nucleic Acid Test
B. Potion(easy version)
C. Potion(hard version)
D. Transition
E. Multigate
F. Angel
G. Brick
H. Hamster and Multiplication
I. Latitude Compressor
J. Palindrome Reversion
K. PTT
L. Chtholly and the Broken Chronograph
M. Super Star Spectacle

Do not open before the contest has started.

Problem A. Nucleic Acid Test

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	128 megabytes

In 2022, the COVID-19(omicron) has been widely spread around the world.

To prevent further spreading, the government decided to restrict people's travel by requiring negative result of *Nucleic Acid Test* for all travellers.

To meet with the restriction, people should take *Nucleic Acid Test* regularly. Now, in your city, there are n buildings, connected by m bidirectional roads. Some of the buildings have *Nucleic Acid Test* station, in which you can take a *Nucleic Acid Test* to get one negative result.

Also, the government announced that a **valid** *Nucleic Acid Test* result should be within t minutes. That is to say, for any time i, you should guarantee that the time of your last *Nucleic Acid Test* is no earlier than $i - t$. You must make sure that you have a **valid** *Nucleic Acid Tests* result everytime you enter a building, or you are not allowed to step in.

You want to visit all the buildings without violating the restriction (i.e. Having a **valid** *Nucleic Acid Test* result at anytime of your trip). You must start at one building with *Nucleic Acid Test* station and take one test at 0-th minute, and also end with one building with test station to take a test for your safety. During the whole trip, you should move with a fixed moving speed v, which means that for a road with length l, the time you go through the road will be $\frac{l}{v}$. Specially, the velocity must be an **integer**.

You wonder what's the minimum valid moving speed for finishing the trip according to the restrictions above.

Input

The first line contains three integers n $(2 \le n \le 300), m \left(0 \le m \le \frac{n(n-1)}{2}\right), k$ $(1 \le k \le n)$, denoting the number of buildings, the number of roads and the number of *Nucleic Acid Test* stations.

The second line contains a single integer t $(0 \le t \le 10^9)$, denoting the biggest time gap between two contiguous *Nucleic Acid Tests*.

Each of the following m lines contains 3 integers a_i, b_i, c_i, $(1 \le a_i, b_i \le n, 1 \le c_i \le 10^9)$. All the roads are bidirectional. Each pair of cities is connected by at most one road. None of the roads connect the same building.

The last line contains k integers s_i, denoting the building number of i-th *Nucleic Acid Test* station. It is guaranteed that when $i \ne j$, $s_i \ne s_j$.

Output

One integer in one line, indicates the smallest valid moving speed.

If no such moving speed exists, print -1 in a single line.

Example

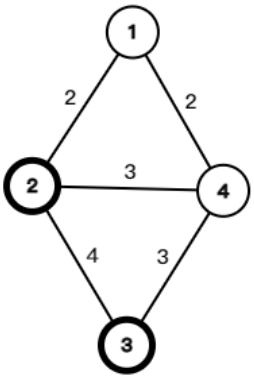
standard input	standard output
4 5 2 3 1 2 2 2 3 4 3 4 3 1 4 2 2 4 3 2 3	2

Note

The valid period of *Nucleic Acid Test* result is 3 minutes.

You can set your moving speed as 2, and start at the *Nucleic Acid Test* station *3*, and then go to building *4*, taking 1.5 minutes. And return to building *2* and take a *Nucleic Acid Test* taking 1.5 minutes.

After that, you can go to building *1* under 1 minute, and return to building *2* under another minute. Totally, your gap this time is 2 minutes, and you end up at a *Nucleic Acid Test* station to confirm that you are safe.



Problem B. Potion(easy version)

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

This is the easy version of the problem C. The only difference between the two versions is that the easy version satisfies a=b=1.

Twilight Sparkle is a Unicorn pony who loves knowledge.

One day, she is learning how to make magic potion with the teaching of Zecora, a zebra living in Everfree Forest.

In the preparatory work, she should mix pure water with the magic water extracted from Everfree Forest in a specific proportion $x : y$. (These two types of liquid can dissolve in each other, of course.)

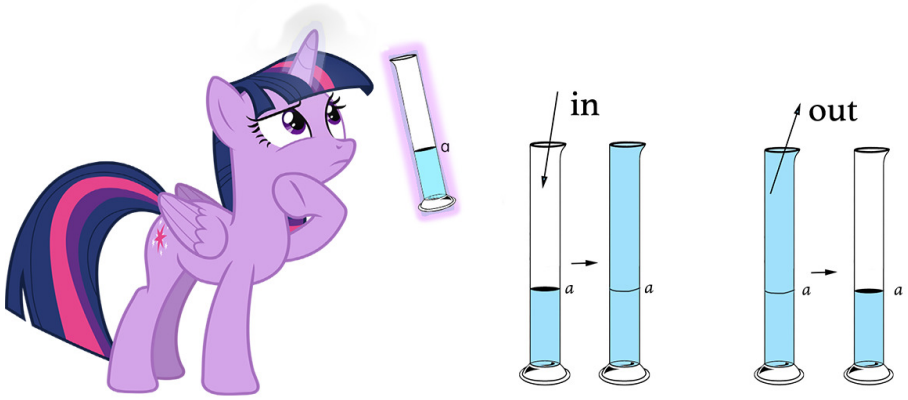
However, she has only one measuring cup with blur scale marks. More specifically, the measuring cup has a volume of $a + b$ ml and only one distinct scale mark at a ml. That is, she can only measures a ml or $a + b$ ml of liquid via this measuring cup precisely.

Now Twilight Sparkle are allowed to perform the following operations any times:

1. Fill the measuring cup with pure water or magic water.

2. Pour the mixture water **out** from the measuring cup and preserve a ml of mixture in the measuring cup.

Twilight Sparkle wonders what is the minimum possible number of **operation 1** to make a mixture with proportion $x : y$ **in the measuring cup**, or tell that is impossible.



Input

The first line contains a single integer T ($1 \le T \le 10^5$) - the number of testcases.

Each testcase contains only one line, four integers x, y, a, b ($1 \le x, y \le 10^{18}, a = b = 1$).

Output

For each testcase, if Twilight Sparkle couldn't make the specific mixture, print a single integer: -1.

Otherwise, print the minimum number of operation 1 to do that.

Example

standard input	standard output
3 3 5 1 1 2 6 1 1 5 7 1 1	4 3 -1

Note

For the first test case, Twilight Sparkle does the following operations:

Fill in **pure** water, then the proportion is $1 : 0$.

Pour out the mixture till a ml remains, and pour in **magic** water, then the proportion is $1 : 1$.

Pour out the mixture till a ml remains, and pour in **pure** water, then the proportion is $3 : 1$.

Pour out the mixture till a ml remains, and pour in **magic** water, then the proportion is $3 : 5$.

For the second test case, Twilight pours in pure, magic and then magic water in order.(And also does operation 2 every time she fills in liquid except the last time.)

For the third test case, there is no way for her to make the specific mixture.

Problem C. Potion(hard version)

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

This is the harder version of the problem B. The only difference between the two versions is that the easy version satisfies a=b=1.

Twilight Sparkle is a Unicorn pony who loves knowledge.

One day, she is learning how to make magic potion with the teaching of Zecora, a zebra living in Everfree Forest.

In the preparatory work, she should mix pure water with the magic water extracted from Everfree Forest in a specific proportion $x : y$. (These two types of liquid can dissolve in each other, of course.)

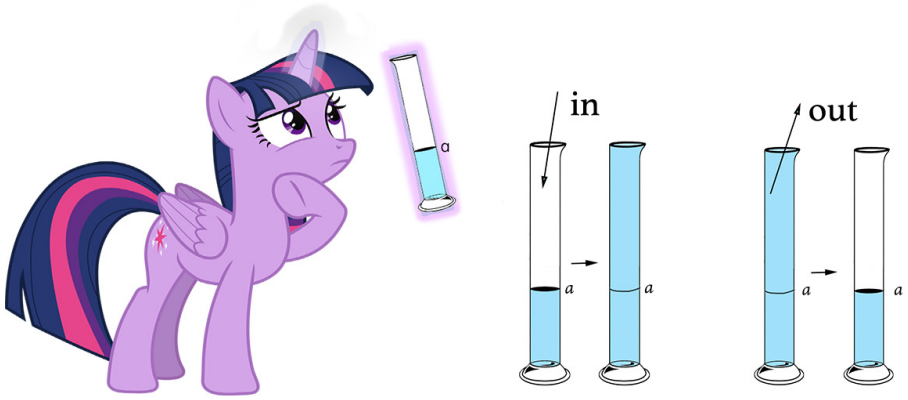
However, she has only one measuring cup with blur scale marks. More specifically, the measuring cup has a volume of $a + b$ ml and only one distinct scale mark at a ml. That is, she can only measures a ml or $a + b$ ml of liquid via this measuring cup precisely.

Now Twilight Sparkle are allowed to perform the following operations any times:

1. Fill the measuring cup with pure water or magic water.

2. Pour the mixture water **out** from the measuring cup and preserve a ml of mixture in the measuring cup.

Twilight Sparkle wonders what is the minimum possible number of **operation 1** to make a mixture with proportion $x : y$ **in the measuring cup**, or tell that is impossible.



Input

The first line contains a single integer T ($1 \le T \le 10^5$) - the number of testcases.

Each testcase contains only one line, four integers x, y, a, b ($1 \le a, b, x, y \le 10^{18}$).

Output

For each testcase, if Twilight Sparkle couldn't make the specific mixture, print a single integer: -1.

Otherwise, print the minimum number of operation 1 to do that.

Example

standard input	standard output
4 3 5 1 1 1 24 1 4 12 113 3 2 8 117 2 3	4 3 -1 4

Note

For the first test case, Twilight Sparkle does the following operations:

Fill in **pure** water, then the proportion is $1 : 0$.

Pour out the mixture till a ml remains, and pour in **magic** water, then the proportion is $1 : 1$.

Pour out the mixture till a ml remains, and pour in **pure** water, then the proportion is $3 : 1$.

Pour out the mixture till a ml remains, and pour in **magic** water, then the proportion is $3 : 5$.

For the second test case, Twilight can fill in pure, magic and then magic water in order. (And also does operation 2 every time she fills in liquid except the last time.)

For the third test case, there is no way for her to make the specific mixture.

For the fourth test case, she can fill in pure, magic, magic and magic water in order.(And also does operation 2 every time she fills in liquid except the last time.)

Problem D. Transition

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

Rarity is a Unicorn pony who is a famous fashion designer.

One day, she is designing an important uniform for Summer Sun Celebration, but the designs of floral patterns on it are always different from her intention. With hard endeavor, she comes up with a perfect idea and can't wait to test it on the uniform!

Now Rarity wants to change the initial pattern a to the new idea b. A pattern can be regarded as a string consisting only of '0' or '1'. It's guaranteed that pattern a and b have the same length.

We define the i-th character of a string s as s_i, and the length of s as $|s|$.

She could do the following operations any times:

1. Choose two indices i, j $(1 \leq i < j \leq |a|)$, then swap a_i and a_j, which costs $j - i$ units of time.

2. Choose an index i $(1 \leq i \leq |a|)$, then flip a_i (i.e. change a_i to 1 if it's 0 initially, vice versa), which costs 1 unit of time.

It's easy for Rarity to find the minimum cost in changing a to b, but how about the number of different legal **operation sets** (i.e. a set consists some operations) to get that in minimum cost?

We consider one operation set legal, if it has at least one operation order to change a to b.

We consider two operation sets different, if and only if one operation exists in one set but not in another.

We consider two operations different, if and only if they choose different index set (i.e. different in size or element).

Since the answer can be quite large, print it modulo $10^9 + 7$.

Input

The first line contains an integers n $(1 \leq n \leq 3 \times 10^5)$ - the length of string a, b.

The second line contains a string of length n - represent as a.

The third line contains a string of length n - represent as b.

Output

Only a single integer - the number of different operation sets modulo $10^9 + 7$.

Examples

standard input	standard output
4 0111 1100	3
7 0101001 1010110	3

Note

We will denote set $\{i, j\}$ as an operation 1 that swaps a_i and a_j, $\{i\}$ as an operation 2 that flips a_i.

For the first test case, the minimum cost is 3, and the possible operation set are $\{\{1, 3\}, \{4\}\}$, $\{\{1, 2\}, \{2, 3\}, \{4\}\}$ and $\{\{1\}, \{3\}, \{4\}\}$.

For the second test case, the minimum cost is 4, and the possible operation set are $\{\{1,2\},\{3,4\},\{5\},\{6,7\}\}$, $\{\{1,2\},\{3\},\{4,5\},\{6,7\}\}$ and $\{\{1\},\{2,3\},\{4,5\},\{6,7\}\}$.

Problem E. Multigate

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Rainbow Dash is a Pegasus pony who loves adventure.

One day, she comes across a sequence of n magic gates, each magic gate has a magic power value a_i and a type b_i.

Initially, Rainbow Dash has a integer x in her hands. When Rainbow Dash passes through the i-th magic gate with a non-negative integer x, x will become x `OR` a_i if b_i is 1 and otherwise x `AND` a_i if b_i is 0, here `OR` and `AND` mean bitwise OR and bitwise AND respectively. Additionally, she must pass through the gate according to the given order.

Since Rainbow Dash has some unicorn friends, she could choose at **most** k magic gates and flip the b_i of these magic gates before she goes. That means, she could flip b_i to 1 if $b_i = 0$ originally and vice versa.

Rainbow Dash needs to maximize the integer x after passing n magic gates. However, she will face this challenge q times **independently**, and each time she is given two non-negative integers x and k, which means the initial number in her hands and the maximum times of flip operations she could use. Please pay attention that the flip operation in one challenge will not influence another challenge.

Input

The first line contains two integers n ($1 \le n \le 2 \times 10^5$) and q ($1 \le n \le 2 \times 10^5$) - the length of array a and the numbers of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < 2^{30}$) - the elements of array a.

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i \le 1$) - the elements of array b.

The next q lines, each line contains two integer x ($0 \le x < 2^{30}$) and k ($0 \le k \le n$) - the initial number and the most flip operation she could use.

Output

For each test case, output a single line containing the maximum possible x after performing at most k operations and passing the magic gates.

Example

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

Note

For the query 1, there is no flip chance and x will pass the following process:

((((6 `AND` 3) `OR` 1)`AND` 3) `AND` 5) = 1.

For the query 2, Rainbow Dash could flip the 4-th magic gate and x will finally become 7, which is maximal.

Query 3 and query 6 is familiar to query 1, which have no flip chance.

Query 4 and query 8 is familiar to query 2, which have to flip the 4-th magic gate to get x maximal.

As for query 5, flipping the 3-rd magic gate and 4-th magic gate would be the best choice.

Problem F. Angel

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Fluttershy is a Pegasus pony who has the reputation of kindness and has unique skill to take care of animals. However, she has a spoiled rabbit pet called Angel, who sometimes makes fun of Fluttershy.

One day, Angel tells Fluttershy that he is hiding in one of the n special holes in Everfree Forest and Fluttershy should find him out, before the midnight!!

To make the game more fair, Angel set some rules.

The n holes straightly lie in one line and Angel gives them indices $1, 2, 3, \ldots, n$ from left to right.

Angel couldn't stay in one hole for more than one minute, and he must move to a hole adjacent to the hole that he currently stays every one minute. That means, if Angel is currently in the i-th hole, he should move to the $i-1$-th hole or $i+1$-th hole in the next minute. Of course, he couldn't move beyond the borders.

Every time **before** Angel moves, Fluttershy could check one of the n holes, and if Angel is in the checked hole at this time, he will be caught.

Help Fluttershy finding a way to catch Angel in minimum number of checks **in the worst case**, or tell her that is impossible.

Attention that we only consider the worst case, that means, after those checks, Angel is bound to be caught no matter how he moves.

Input

Only a single integer n ($1 \le n \le 10^3$) - the number of holes.

Output

If Fluttershy could never catch Angel, print a single integer: -1.

Otherwise, print two lines:

The first line contains a single integer m - the minimal number of checks Fluttershy has to make to catch Angel in the worst case.

The second line contains m integers $a_1, a_2, \ldots, a_m$ - the index of the hole in i-th check.

Example

standard input	standard output
3	2 2 2

Note

If Angel was initially in hole 2, Fluttershy would catch him in check 1.

Otherwise, Fluttershy would catch him in check 2, Angel would be in hole 2 at that time since he had to move from hole 1 or hole 3.

It can be proved that Fluttershy has no way to catch Angel in only one check, in the worst case.

Problem G. Brick

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Apple Jack is an Earth pony who lives and works at Sweet Apple Acres and Pinkie Pie is an energetic Earth pony who loves party!!

One day, Pinkie Pie destroyed the wall around Sweet Apple Acres accidentally. To restore it rapidly by bricks, she summoned her friends for help.

With the endeavor of ponys, they simplified the problem in a formal way.

They considered the wall in ruins as an integer array $\{h_i\}$ of length n, and bricks of scale 1×2 can be placed horizontally or vertically on it.

More specifically, one brick could:

1. Add 2 to any h_i.

2. Add 1 to both h_i and h_{i+1} if $h_i = h_{i+1}$.

We define that the wall is restored when there exists an integer x satisfying $h_i = x$ for all i.

Now ponys want to know, what's the minimal possible height x for a restored wall, or tell them it's impossible to restore the wall.

Input

The first line contains an integers n ($1 \le n \le 2 \times 10^5$) - the length of array h.

The second line contains n integers $h_1, h_2, \ldots, h_n$ ($1 \le h_i \le 10^9$) - the elements of array h.

Output

If the wall can't be restored, print a single integer: -1.

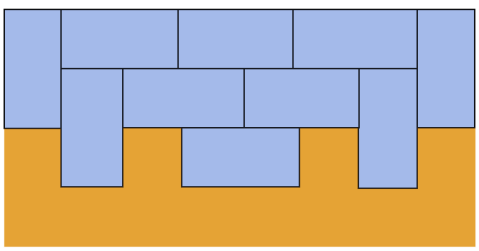
Otherwise, print an integer x - the minimal height of the wall.

Example

standard input	standard output
8 2 1 2 1 1 2 1 2	4

Note

For the example we have a figure for explanation.



Problem H. Hamster and Multiplication

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

ZSGW is a cute hamster studying in Wuhan Hamster University (A.k.a. WHU). He finds it difficult to learn math courses such as advanced mathematics, because he even misunderstands the definition of multiply operation!

ZSGW can only correctly multiply numbers when the result has only one digit (like $2 \times 4 = 8$). When the result has multiple digits (i.e. the result is not less than 10), for example, calculating 7×2, things would get wrong. We all know the result should be 14, but hamster will regard 14 as 1×4, so the result of 7×2 would be 4. He will repeat this procedure until the result has only one digit, and this is how ZSGW calculate 8×9: $8 \times 9 = 72 = 7 \times 2 = 14 = 1 \times 4 = 4$.

The reason is that ZSGW cannot understand numbers larger than 9. For numbers with multiple digits, ZSGW will regard the number as the result of multiplying the digits together (according to his own definition of multiplication). Formally, let $f(x)$ be the value that ZSGW thinks the x is. Then here is an example:

$f(266) = f(2 \times 6 \times 6) = f(72) = f(7 \times 2) = f(14) = f(1 \times 4) = f(4) = 4$

Here is a more precise definition of $f(x)$:

$$f(x) = \begin{cases} x & (x < 10) \\ f(\prod_i x_i) & (x \geq 10) \end{cases}$$

Where x_i refers to the i-th digit of x.

Now here comes the final question: Given a number n, you need to calculate the sum of this function from 1 to n.

That is: Calculate $\sum_{i=1}^{n} f(i)$.

Input

A single number n ($1 \leq n < 10^{18}$) whose description is on the above.

Output

A single number *ans* in a line as the answer.

Examples

standard input	standard output
5	15
22	96
522	1788
522522	366166

Note

$f(266) = f(2 \times 6 \times 6) = f(72) \neq f(f(12) \times 6)$

Problem I. Latitude Compressor

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

"Our world is surrounded by the pale, 6000 kilometres to the north, and even more to the south, east and west. In the Dolorian-era, we discovered the New New World, the piece of reality we're standing on."

"In modern times, it is possible to force dimensions on the pale – we can even compress its latitude, with a permutation generator, bouncing radio waves from one end to the other. Shortening the path."

Now, you got a latitude compressor and you want to evaluate its effect, you can hardly tell how it works, but you know, once you sent a permutation p, you will get the permutation p^m (it means the product of m same permutations), m nanoseconds later.

After sending all the permutations with length n, its effect can be evaluated as the number of different permutations after compressed by the compressor.

Since the answer maybe too large, you just need to output it modulo 998244353.

By the way, if you don't know what permutation and the product of permutations is, here is a brief introduction :

A permutation with length n can be represented as :

$$f = \begin{pmatrix} 1 & 2 & \dots & n \\ f(1) & f(2) & \dots & f(n) \end{pmatrix}$$

where $f(1), f(2), \dots, f(n)$ are integers in $[1, n]$ and pairwise different.

And the product of two permutation is defined as :

$$f \times g = \begin{pmatrix} 1 & 2 & \dots & n \\ g(f(1)) & g(f(2)) & \dots & g(f(n)) \end{pmatrix}$$

Input

One line contains two integers n and m ($1 \le n, m \le 50000$).

Output

Print a single integer that represents the answer *modulo 998244353*.

Examples

standard input	standard output
3 2	3
4 2	12
6 12	145
654 72	922925108

Note

For the first test case, we have $n = 3,\ m = 2$.

There are 6 different permutations with length 3, we present it as $f = (f(1), f(2), f(3))$, and there square will be:

$f = (1, 2, 3), \quad (1, 2, 3) \xrightarrow{f} (1, 2, 3) \xrightarrow{f} (1, 2, 3)$

$f = (1, 3, 2), \quad (1, 2, 3) \xrightarrow{f} (1, 3, 2) \xrightarrow{f} (1, 2, 3)$

$f = (2, 1, 3), \quad (1, 2, 3) \xrightarrow{f} (2, 1, 3) \xrightarrow{f} (1, 2, 3)$

$f = (2, 3, 1), \quad (1, 2, 3) \xrightarrow{f} (2, 3, 1) \xrightarrow{f} (3, 1, 2)$

$f = (3, 1, 2), \quad (1, 2, 3) \xrightarrow{f} (3, 1, 2) \xrightarrow{f} (2, 3, 1)$

$f = (3, 2, 1), \quad (1, 2, 3) \xrightarrow{f} (3, 2, 1) \xrightarrow{f} (1, 2, 3)$

Then we got 3 different permutations after compressed.

Problem J. Palindrome Reversion

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In computer science, a one-way function is a function that is easy to compute on every input, but hard to invert given the image of a random input, they are fundamental tools for cryptography.

But the following is not:

Given a string s, can you reverse exactly an interval in s and make s a palindrome?

If you don't know what is palindrome and reversion, here is a brief definition:

A palindrome is a string which reads the same backward as forward.

And for a string $s = s_1 s_2 \dots s_{l-1} s_l s_{l+1} \dots s_{r-1} s_r s_{r+1} \dots s_n$,

we define $\text{reverse}(s, l, r) = s_1 s_2 \dots s_{l-1} s_r s_{r-1} \dots s_{l+1} s_l s_{r+1} \dots s_n$

Input

One line contains a string s, consisting of lowercase English letters, it's guaranteed that $1 \le |s| \le 10^5$.

Output

Print "-1 -1" if there's no solution.

Otherwise output two integers l, r $(1 \le l \le r \le |s|)$ in a line representing the interval. If there are multiple solutions, just output any of them.

Examples

standard input	standard output
bcbaa	1 4
ababaa	3 6
abcb	-1 -1

Note

"bcbaa" → "abcba"

"ababaa" → "abaaba"

Problem K. PTT

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In Arcaea, there's a value called Potential (often abbreviated as "PTT") to measure a player's game level. It acts as a representative value of the player's best and recent performances.

Here, you'll learn how to calculate a player's potential score of a single gameplay.

There are two relevant quantities to the potential, each explained in its own section.

Chart Constant

The Chart Constant is an internal value assigned to a chart, which is a value between 1.0 and 11.5 with one decimal place. This value reflects the difficulty of a chart. However,some angry players believe that the unreasonable last decimal was rolled with dice.

Score Modifier

The Score Modifier is a value computed from your score in a round. It can be positive or negative, depending on whether your score is above or below 9,500,000.

Below is a table showing the formula for calculating the score modifier for various score ranges.

Score	Score Modifier
$\geq 10\,000\,000$	2.0
$9\,800\,000 - 9\,999\,999$	$1.0 + (Score - 9\,800\,000)/200\,000$
$\leq 9\,800\,000$	$(Score - 9\,500\,000)/300\,000$

Play Rating

And now you are able to calculate the rating of one single gameplay. Just add the chart constant with score modifier, and **set it to zero if it's below zero.** We call it play rating.

Now you are given T groups of gameplay with score and chart constant, please calculate the play rating of each gameplay.

Input

The first line contains one integer T ($1 \leq T \leq 100000$) denoting the number of gameplays.

Each of the following T lines contains one integer n ($0 \leq n \leq 10001576$), and one number with one decimal place c ($1.0 \leq c \leq 11.5$), denoting the score and chart Constant of one single gameplay.

Output

Output T lines, each line with a real number denoting a play rating. Your answer is considered correct if the absolute or relative error with standard output doesn't exceed 10^{-6}.

Example

standard input	standard output
3 10001343 9.5 9988535 11.3 9794071 11.0	11.5000000 13.2426750 11.9802367

Problem L. Chtholly and the Broken Chronograph

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Chtholly gives you an array of n elements, the i-th of which is a_i.

Each element in the array has an independent state s_i, where $s_i = 0$ denotes the i-th element is blocked, and $s_i = 1$ denotes it is activated.

In order to maintain the array, Chtholly needs you to perform q operations, and there are four kinds of them:

1 x: Block element x, i.e. change s_x to 0. It's guaranteed that the element is activated before the operation.

2 x: Activate element x, i.e. change s_x to 1. It's guaranteed that the element is blocked before the operation.

3 l r x: Add x to each activated element in interval $[l, r]$, i.e. for each i such that $l \le i \le r$ and $s_i = 1$, assign $a_i + x$ to a_i.

4 l r: Print the sum of elements in interval $[l, r]$. Note that this operation is irrelevant to the current states of elements.

Input

The first line of the input contains two integers $n, q\,(1 \le n, q \le 10^5)$.

The second line contains n integers, the i-th of which is $a_i\,(1 \le a_i \le 10^8)$.

The second line contains n integers, the i-th of which is $s_i\,(s_i \in \{0, 1\})$.

The next q lines, each line describe an operation. The forms of the operations are described in the statements above.

It is guaranteed that for each operation of type 3 or 4, $1 \le l \le r \le n$, and for each operation of type 3, $1 \le x \le 10^8$.

Output

For each operation of type 4, output one line containing the answer.

Example

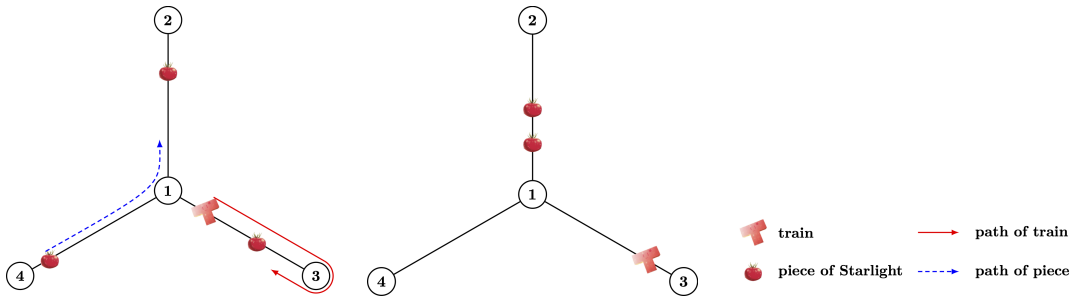
standard input	standard output
4 8 4 2 5 3 1 0 0 1 2 3 3 1 4 1 1 3 4 1 4 1 1 2 2 3 1 3 2 4 1 4	17 19

Problem M. Super Star Spectacle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Karen is left alone in a solitary train car heading through a vast desert, on her journey of finding herself.

Rails in the desert can be represented as an undirected tree (i.e., have $n-1$ railway sections connecting n vertices). The glitter of Starlight splits into pieces on the rails, and Karen needs to arrange some trains to collect them **all**. Trains scan the rails by moving **continuously** along the rails and picking up pieces encountered. But pieces do move as well, even much faster. Here is a possible movement of some pieces and trains.



The train will absolutely reach the next station, but it is questioning, both for the stages and girls. Pieces of Starlight are **too tiny to locate**, so it takes a number of trains and careful work to **ensure no pieces are left**. Please determine the minimum number of trains needed, to send Karen to the final Revue, and to send Starlight onto the next stage.

Input

The first line contains a single integer n ($1 \le n \le 3 \times 10^5$), which represents the number of vertices. Vertices are indexed from 1 to n.

Each of the following $n-1$ lines will contain two integers x_i, y_i ($1 \le x_i, y_i \le n$), where x_i and y_i are indices of the vertices connected by the i-th railway section.

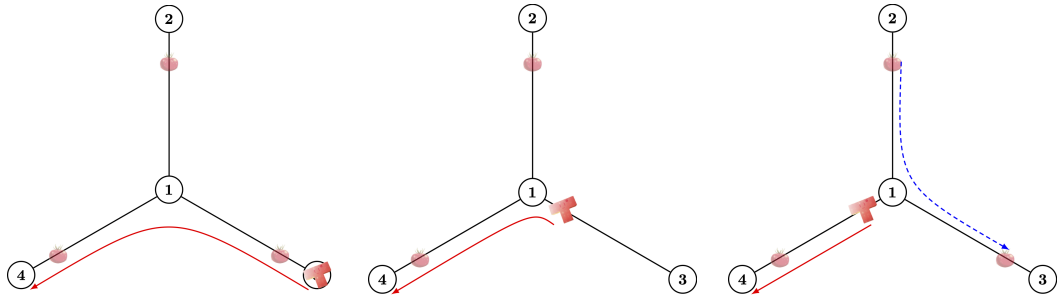
Output

Print a single integer that represents the minimum number of trains needed.

Example

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

Note



In the example given, it's impossible to collect every piece with a single train. As the graphs show, the edge between (1) and (3) scanned by the train temporarily turns clear (i.e. no pieces on it). But there might always be some pieces entering it, and you could not find them. In that case, you could never determine that no pieces are left.

It's easy to construct a solution with 2 trains.