

icpc international collegiate programming contest

ICPC North America Regionals 2019

ICPC Southeast USA
Regional Contest

Official Problem Set

Division 2



icpc.foundation



icpc global
programming
tools sponsor



TWO SIGMA

icpc
north
america
sponsor



ICPC Southeast USA Regional Contest

Division 2

Balanced Animals..... 3

Carryless Square Root 5

Checkerboard 6

From A to B..... 7

Elven Efficiency 8

Even or Odd? 10

Glow, Pixel, Glow!..... 11

Jumping Path 13

Levenshtein Distance 16

Rainbow Strings 17

ReMorse 18

Hosted by:

- College of Charleston**
- Florida International University**
- Kennesaw State University**
- University of West Florida**



ICPC Southeast USA Regional Contest

Balanced Animals

Time limit: 1 second

To save money, Santa Claus has started hiring other animals besides reindeer to pull his sleigh via short term contracts. As a result, the actual animals that show up to pull his sleigh for any given trip can vary greatly in size.

Last week he had 2 buffalo, 37 voles and a schnauzer. Unfortunately, both buffalo were hitched on the left side and the entire sleigh flipped over in mid-flight due to the weight imbalance.

To prevent such accidents in the future, Santa needs to divide the animals for a given trip into two groups such that the sum of the weights of all animals in one group equals the sum of the weights of all animals in the other. To make the hitching process efficient, Santa is seeking an integer target weight t such that all animals that are lighter than t go in one group and those heavier than t go into the other. If there are multiple such t , he wants the smallest one. There's one small wrinkle: what should be done if there some animals have weight exactly equal to t ? Santa solves the problem this way: if there are an even number of such animals, he divides them equally among the two groups (thus distributing the weight evenly). But if there are an odd number of such animals, then one of those animals is sent to work with the elves to make toys (it is not put in either group), and the remaining (now an even number) are divided evenly among the two groups.

Input

The first line contains an integer n ($2 \leq n \leq 10^5$), indicating the number of animals.

Each of the next n lines contain a positive integer w ($1 \leq w \leq 2 \cdot 10^5$). These are the weights of the animals (in ounces).

Output

Output the smallest integer target weight t , as described above. It's guaranteed that it is possible to find such an integer.



ICPC Southeast USA Regional Contest

Sample Input	Sample Output
4 3 6 1 2	4
4 11 8 3 10	10
2 99 99	99

ICPC Southeast USA Regional Contest

Carryless Square Root

Time limit: 1 second

Carryless addition is the same as normal addition, except any carries are ignored (in base 10). Thus, $37 + 48$ is 75, not 85.

Carryless multiplication is performed using the schoolbook algorithm for multiplication, column by column, but the intermediate sums are calculated using *carryless* addition. Thus:

$$\begin{aligned} 9 \cdot 1234 &= 9000 + (900 + 900) + (90 + 90 + 90) + (9 + 9 + 9 + 9) \\ &= 9000 + 800 + 70 + 6 = 9876 \end{aligned}$$

$$90 \cdot 1234 = 98760$$

$$99 \cdot 1234 = 98760 + 9876 = 97536$$

Formally, define c_k to be the k^{th} digit of the value c . If $c = a \cdot b$ then

$$c_k = \left[\sum_{i+j=k} a_i \cdot b_j \right] \bmod 10$$

Given an integer n , calculate the smallest positive integer a such that $a \cdot a = n$ in *carryless* multiplication.

Input

The input consists of a single line with an integer n ($1 \leq n \leq 10^{25}$).

Output

Output the smallest positive integer that is a *carryless* square root of the input number, or -1 if no such number exists.

Sample Input

Sample Output

6	4
149	17
123476544	11112
15	-1

ICPC Southeast USA Regional Contest

Checkerboard

Time limit: 1 second

An $r \times c$ grid of squares is to be colored in a checkerboard style. The board will be filled with rectangles made up of the grid squares. The heights and widths of the rectangles will be specified. **Black** and **White** are the only two colors of the rectangles. Any two adjacent rectangles that share a side should be colored differently. The top-left rectangle should be **Black**. Print the checkerboard.

Input

The first line contains four space-separated integers r , c , v and h ($1 \leq v \leq r \leq 50$, $1 \leq h \leq c \leq 50$) where the checkerboard is to have r rows and c columns, with v rectangles vertically and h rectangles horizontally.

Each of the next v lines contain a single positive integer a . The sum of the a values will be exactly r . These are the heights of the v rectangles in each column, in order from top to bottom.

Each of the next h lines contain a single positive integer b . The sum of the b values will be exactly c . These are the widths of the h rectangles in each row, in order from left to right.

Output

Print the described checkerboard, in the form of r strings of length c , one per line. The strings should only contain the characters upper-case **B** (for a **Black** square) and upper-case **W** (for a **White** square).

Sample Input	Sample Output
6 5 3 2 1 2 3 3 2	BBBWW WWWBB WWWBB BBBWW BBBWW BBBWW
4 4 2 2 1 3 3 1	BBBW WWWB WWWB WWWB

ICPC Southeast USA Regional Contest

From A to B

Time limit: 1 second

You are given two integers, a and b . You want to transform a into b by performing a sequence of operations. You can only perform the following operations:

- Divide a by two (but only if a is even)
- Add one to a

What is the minimum number of operations you need to transform a into b ?

Input

The single line of input contains two space-separated integers a and b ($1 \leq a, b \leq 10^9$)

Output

Output a single integer, which is the minimum number of the given operations needed to transform a into b .

Sample Input

Sample Output

103 27	4
3 8	5

ICPC Southeast USA Regional Contest

Elven Efficiency

Time limit: 5 seconds

Like many creatures featured in programming problems, the animals of the forest love playing games with stones. They recently came up with a game to teach the younger animals about divisibility. In this game, each animal starts with a pile of stones. At the start of the game, a series of numbers is called out. For each number that is called, every animal whose number of stones is divisible by the called number scores a point. At the end of the game, the animal with the most points wins.

Emma the forest elf has watched the forest animals play this game many times, and has grown tired of watching the winning animal gloat about how many points they scored. To prevent this from happening, she plans to meddle in the next game the animals play to ensure that no animal scores any points. She plans to wait atop a nearby tree, and keep track of how many stones each animal has. Each round, if an animal is about to score a point, she can toss a stone into that animal's pile, increasing their number of stones by one. The tossed stone stays in that pile for the rest of the game. Throughout the course of the game, she may need to toss several stones into the same pile. But stones are heavy, and she wants to carry as few as possible to the top of her hideout tree. She already knows how many stones each of the n animals will start with, as well as the number to be called out in each of the m rounds of the game, but she wants you to calculate the minimum total number of stones she will have to throw to ensure that no animal scores any points.

Input

The first line of input contains two space-separated integers n and m ($1 \leq n, m \leq 10^5$), where n is the number of animals, and m is the number of rounds of the game.

Each of the next n lines contain a single integer a ($1 \leq a \leq 3 \cdot 10^5$), which are the numbers of stones held by each animal.

Each of the next m lines contain a single integer k ($2 \leq k \leq 3 \cdot 10^5$), which are the numbers called out, in order.

Output

Output a single integer, which is the minimum number of stones that Emma must use to prevent any and all animals from scoring any points.



ICPC Southeast USA Regional Contest

Sample Input

```
3 5
10
11
12
2
11
4
13
2
```

Sample Output

```
12
```

ICPC Southeast USA Regional Contest

Even or Odd?

Time limit: 1 second

Your friend has secretly picked n consecutive positive integers between 1 and 10^{18} and wants you to guess if their sum is even or odd.

If the sum must be even, write **2**. If the sum must be odd, write **1**. If the sum could be even or could be odd, write **0**.

Input

The single line of input contains a single integer n ($1 \leq n \leq 10^9$).

Output

Output **2** if the sum of any n consecutive integers in the range from 1 to 10^{18} must be even, **1** if the sum must be odd, or **0** if the sum could be either even or odd.

Sample Input	Sample Output
3	0
6	1
12	2

ICPC Southeast USA Regional Contest

Glow, Pixel, Glow!

Time limit: 2 seconds

An LCD panel is composed of a grid of pixels, spaced **1 ALU** (Arbitrary Length Unit) apart both horizontally and vertically. Wires run along each row and each column, intersecting at the pixels. Wires are numbered beginning with **1** and proceeding up to a panel-dependent maximum. The vertical wire numbered **1** lies along the left edge of the panel, and the horizontal wire numbered **1** lies along the bottom edge of the panel.

For a period of time, pulses of current will be sent down selected wires. The current flows down the wires at a speed of one *ALU* per *ATU* (Arbitrary Time Unit). The pulses themselves have a length measured in *ATUs*. A pixel will activate, and glow, when current is passing through both intersecting wires at the same time. It will deactivate when either current is no longer present. If the leading edge of a pulse on one wire reaches the intersection at the exact same time that the trailing edge of a pulse on the other wire leaves that intersection, the pixel is not activated.

All pulses in vertical wires start from the bottom of the grid. All pulses in horizontal wires start from the left of the grid. At most one pulse will travel along any one wire.

Given the schedule of pulses to be sent through the wires, determine how many pixels will have been activated by the time all pulses have exited the top and right of the grid.

Input

The first line of input contains a single integer n ($1 \leq n \leq 2 \cdot 10^5$), which is the number of current pulses.

Each of the next n lines consists of four space-separated parameters, describing a pulse:

d	character	($d = 'h'$, or $d = 'v'$)
t	integer	($1 \leq t \leq 2 \cdot 10^5$)
m	integer	($1 \leq m \leq 2 \cdot 10^5$)
a	integer	($1 \leq a \leq 10^5$)

where d indicates whether the pulse is horizontal ('h') or vertical ('v'), t is the time at which it starts (i.e., when a horizontal [vertical] pulse crosses vertical [horizontal] wire #1), m is its length, and a is the wire (horizontal or vertical) along which the pulse will travel.



ICPC Southeast USA Regional Contest

Output

Output a single integer, which is the number of pixels that will have activated by the time the last pulse of current has left the grid.

Sample Input**Sample Output**

4 h 1 4 1 v 2 4 2 h 10 2 2 v 11 2 3	2
4 h 1 10 1 h 5 10 2 v 1 10 1 v 5 10 3	4
7 v 1 3 1 v 1 15 2 h 4 5 1 h 5 5 2 h 6 5 3 h 7 5 4 h 8 5 5	5

ICPC Southeast USA Regional Contest

Jumping Path

Time limit: 10 seconds

You are given a rooted tree where each vertex is labeled with a non-negative integer.

Define a *Jumping Path* of vertices to be a sequence of vertices v_1, v_2, \dots, v_k where v_i is an ancestor of v_j for all $i < j$. Note that v_i is an ancestor of v_{i+1} , but not necessarily the parent of v_{i+1} (hence the *jumping* part of a *jumping path*).

Compute two quantities:

- The length (number of vertices) of the longest *jumping path* where the labels of the vertices are nondecreasing.
- The number of *jumping paths* of that length where the labels of the vertices are nondecreasing.

Input

The first line of input contains an integer n ($1 \leq n \leq 10^6$), which is the number of vertices in the tree. Vertices are numbered from 1 to n , with vertex 1 being the tree root.

Each of the next n lines contains an integer x ($0 \leq x \leq 10^6$), which are the labels of the vertices, in order.

Each of the next $n - 1$ lines contains an integer p ($1 \leq p \leq n$), which are the parents of nodes 2 through n , in order.

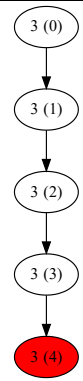
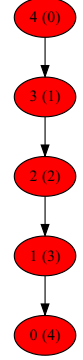
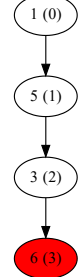
It is guaranteed that the vertices form a single tree, i.e., they are connected and acyclic.

Output

Output a single line with two integers separated by a space.

The first integer is length of the longest *jumping path* where the labels of the vertices are nondecreasing. The second integer is the number of *jumping paths* of that length where the labels of the vertices are nondecreasing. As the second integer may be large, give its value modulo 11092019.

ICPC Southeast USA Regional Contest

Sample Input	Sample Output	Diagram
5 3 3 3 3 1 2 3 4	5 1	
5 4 3 2 1 0 1 2 3 4	1 5	
4 1 5 3 6 1 2 3	3 2	



ICPC Southeast USA Regional Contest

6 1 2 3 4 5 6 1 1 1 1 1 1	2 5	
---	-----	--

ICPC Southeast USA Regional Contest

Levenshtein Distance

Time limit: 1 second

The *Levenshtein Distance* between two strings is the smallest number of simple one-letter operations needed to change one string to the other. The operations are:

- Adding a letter anywhere in the string.
- Removing a letter from anywhere in the string.
- Changing any letter in the string to any other letter.

Given a specific alphabet and a particular query string, find all other unique strings from that alphabet that are at a *Levenshtein Distance* of 1 from the given string, and list them in alphabetical order, with no duplicates.

Note that the query string must not be in the list. Its *Levenshtein Distance* from itself is 0, not 1.

Input

Input consists of exactly two lines. The first line of input contains a sequence of unique lower-case letters, in alphabetical order, with no spaces between them. This is the alphabet to use.

The second line contains a string s ($2 \leq |s| \leq 100$), which consists only of lower-case letters from the given alphabet. This is the query string.

Output

Output a list, in alphabetical order, of all strings which are a *Levenshtein Distance* of 1 from the query string s . Output one word per line, with no duplicates.

Sample Input

```
eg  
egg
```

Sample Output

```
eeg  
eegg  
eg  
ege  
egeg  
egge  
eggg  
gegg  
gg  
ggg
```


ICPC Southeast USA Regional Contest

Rainbow Strings

Time limit: 1 second

Define a *Rainbow String* as a string where every letter in the string is distinct. The empty string is a *Rainbow String*.

Given a string of lower-case letters, compute the number of different subsequences which are *Rainbow Strings*. Two subsequences are different if letter *at a specific position* is included in one subsequence but not the other. Thus, two different subsequences may result in the same string.

For example, consider the string `aab`. The following six subsequences (in bold and underlined) are the only *Rainbow Strings* in `aab`:

a`ab` `a`**a**`b` `a`**a**`b` **a**`a`**b** `a`**a****b** **<empty>**

The answer may be large, so output the answer modulo 11092019.

Input

The single line of input contains a string s ($1 \leq |s| \leq 10^5$) which consists only of lower-case letters.

Output

Output a single integer, which is the number of subsequences of s which are *Rainbow Strings*.

Sample Input

Sample Output

aab	6
icpcprogrammingcontest	209952

ICPC Southeast USA Regional Contest

ReMorse

Time limit: 1 second

Morse code is an assignment of sequences of dot and dash symbols to alphabet characters. You are to reassign the sequences of Morse code so that it yields the shortest total length to a given message, and return that total length.

A dot symbol has length 1. A dash symbol has length 3. The gap between symbols within a character encoding has length 1. The gap between character encodings has length 3. Spaces, punctuation, and alphabetic case are ignored, so the text

The quick brown dog jumps over the lazy fox.

is encoded as though it were

THEQUICKBROWNDOGJUMPSOVERTHELAZYFOX

For instance, for the input **ICPC**, the answer is 17. Encode the **C**s with a single dot, the **I** with a dash, and the **P** with two dots, for an encoding of

—

which has length $(3) + 3 + (1) + 3 + (1 + 1 + 1) + 3 + (1) = 17$.

Input

The single line of input consists of a string s ($1 \leq |s| \leq 32000$) of upper-case or lower-case letters, spaces, commas, periods, exclamation points, and/or question marks. Everything but the letters should be ignored. The line will contain at least one letter.

Output

Output a single integer, which is the length of s when encoded with an optimal reassignment of the sequences of Morse code.



ICPC Southeast USA Regional Contest

Sample Input

Sample Output

ICPC	17
A	1
The quick brown dog jumps over the lazy fox.	335