# Problem A - Adders

Time limit: 0.2 s
Memory limit: 512 MiB

Ivan is bored during a lecture on venomous snakes and is trying to add together all the positive divisors of 100. Of course, the correct result is $f(100) = 1+2+4+5+10+20+25+50+100 = 217$.

Let $f(k)$ be the sum of all the positive divisors of $k$. Given integers $a$ and $b$, find the sum of all $f(k)$, for $k$ between $a$ and $b$ inclusive.

## Input

The first line contains two integers $a$ and $b$ ($1 \leq a \leq b \leq 10^9$).

## Output

Output a single integer – the sum $f(a) + f(a+1) + ... + f(b)$.

## Example

| input | input |
|---|---|
| `100 100` | `1 10` |
| **output** | **output** |
| `217` | `87` |

# Problem B - Battle

Time limit: 1 s
Memory limit: 512 MiB

Battle Checkers is a single-player game played on a $n$-by-$n$ chessboard with $n$ figures. The rows are denoted with integers 1 through $n$, top to bottom, the columns with integers 1 through $n$, left to right. The figures are denoted with integers 1 through $n$.

In each step, the player can move one figure to an empty neighboring square left, right, up or down. The goal of the game is to rearrange the figures into a configuration where each row and each column contains *exactly one figure*.

Given an initial position, find a way to win the game using the minimal number of steps.

## Input

The first line contains an integer $n$ ($3 \leq n \leq 500$) – the size of the board (and the number of figures). The $k$-th of the following $n$ lines contains two integers $r_k$ and $c_k$ ($1 \leq r_k, c_k \leq n$) – the row and the column of figure $k$. No two figures have the same location.

## Output

The first line should contain a single integer $m$ – the minimal number of steps needed to win the game. The following $m$ lines should contain the steps that lead to a win (in chronological order).

Each of those lines should contain an integer $b$ and a character $d$ – the index of the figure to move (between 1 and $n$ inclusive) and the direction to move it in. The direction is one of the uppercase letters `L`, `R`, `U` or `D` meaning left, right, up or down, respectively.

If there are more solutions, output any one.

## Example

**input**

```
5
1 1
1 2
1 3
1 4
1 5
```

**output**

```
10
5 D
5 D
5 D
5 D
4 D
4 D
4 D
3 D
3 D
2 D
```

**input**

```
6
1 1
1 2
2 1
5 6
6 5
6 6
```

**output**

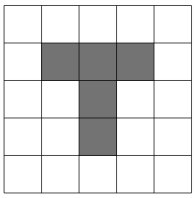
```
8
4 U
5 U
3 D
2 D
5 L
4 L
2 R
3 R
```

# Problem C - Cubes

Time limit: 1 s
Memory limit: 512 MiB

Five cubes and a robot are scattered in a coordinate plane. The robot and each of the cubes fully occupies one unit square whose corners are points with integer coordinates. Their locations are given by specifying the coordinates ($x$ and $y$) of the lower-left corner. As usual, the $x$ coordinate grows to the right while the $y$ coordinate grows upwards. Cubes are denotes with integers 1 through 5.

The robot is remote controlled and, in each step, can be commanded to move one unit square up, down, left or right. If the robot tries to move onto a square occupied by a cube, it pushes the cube onto the next unit square in the direction of the movement. When two cubes touch along a side, they become permanently interlocked and function as one *bundle*. Of course, bundles can grow and contain more cubes. When the robot pushes a cube which is a part of a bundle, the whole bundle moves together without rotation.



The robot is initially located in the origin (both $x$ and $y$ coordinates are 0). Given the locations of the five cubes, find a sequence of commands for the robot that moves the cubes into a T-shaped formation like in the figure above. The exact location of the cubes is arbitrary but the letter T has to be oriented correctly. The length of the sequence does not have to be minimal – any sequence of up to 10 000 commands will be accepted.

## Input

The input consists of five lines. The $k$-th line contains two integers $x_k$ and $y_k$ ($-5 \leq x_k, y_k \leq 5$) – the location of cube $k$.

No two cubes will be at the same location and no two cubes will touch along a side. The origin will be empty.

## Output

Output a single line containing a string consisting of at most 10 000 characters, representing commands to be given in chronological order. Each character is one of the uppercase letters `L`, `R`, `U` or `D` meaning left, right, up or down, respectively.

## Example

**input**

```
0 1
-1 0
1 0
0 -1
0 -3
```

**output**

```
DRRUUULLDD
```

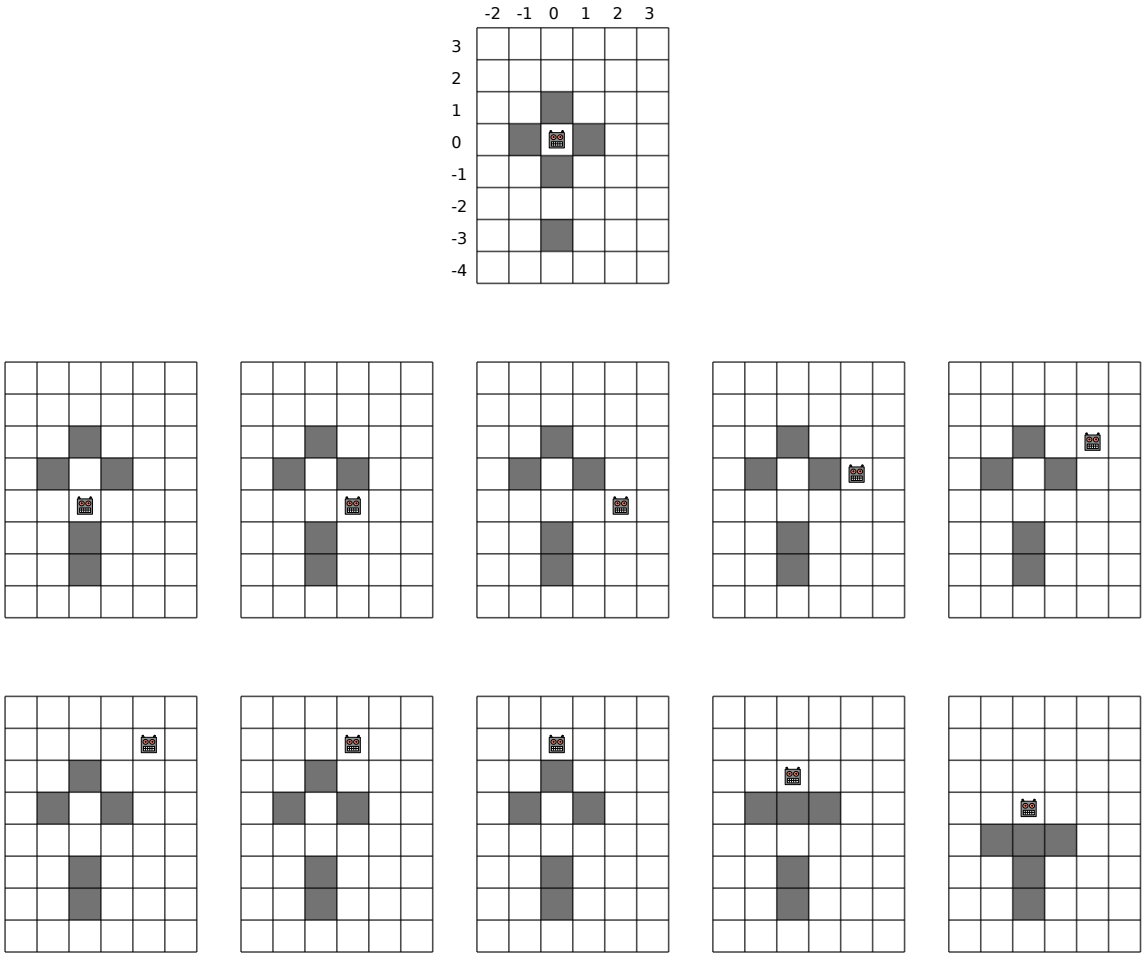
**input**

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

**output**

```
URRDLLURUULDDLLLDR
```

First example input and the corresponding output are depicted below.



# Problem D - Dwarfs

Time limit: 1 s
Memory limit: 512 MiB

Every day, while the dwarfs labour in the mines, Snow White cleans their house and prepares for dinner – seven chairs, seven plates, seven forks and seven knives for the seven hungry dwarfs.

One day, to her astonishment, nine dwarfs returned from the mines – all claiming to be one of the original seven. Fortunately, every dwarf has a unique positive integer written on his hat and the Snow White knows that the numbers of her seven dwarfs add up to exactly 100.

Find the seven dwarfs whose numbers add up to exactly 100. The test data will be such that there is always a unique solution.

## Input

The input consists of nine lines, each containing a single integer between 1 and 99 inclusive. All numbers will be different.

## Output

Output seven lines, each containing a single integer from the input (in arbitrary order). The numbers should be different and add up to exactly 100.

## Example

**input**

```
7
8
10
13
15
19
20
23
25
```

**output**

```
7
8
10
13
19
20
23
```

**input**

```
8
6
5
1
37
30
28
22
36
```

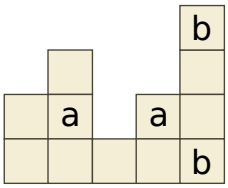
**output**

```
8
6
5
1
30
28
22
```

# Problem E - Elements

Time limit: 4 s
Memory limit: 512 MiB

Young chemist Luka has decided to construct his own periodic table of elements. The table consists of $n$ columns of varying heights aligned along the bottom, as depicted in the figure below.




After drawing the table, Luka has started to fill it with chemical elements. First, he wants to pick locations for the $k$ noble gases, and he does not want them lined up. Formally, we say that two squares of the board are *lined up* if they are either in the same column or in the same row and, additionally, the straight line segment connecting them lies fully inside the table.

In the figure above, the two squares denotes with the letter `b` are lined up while the two denoted with the letter `a` are not.

Given the layout of the table, we are interested in the total number of ways to choose $k$ locations in the table so that no two are lined up. Find that number modulo $10^9 + 7$.

## Input

The first line contains two integers $n$ and $k$ ($1 \leq n, k \leq 500$) – the number of columns and the number of noble gases. The following line contains $n$ positive integers smaller than 1 000 000 – the heights of the columns left to right.

## Output

Output a single integer – the number of ways to choose the locations modulo $10^9 + 7$.

## Example

| input | input | input |
|---|---|---|
| 3 3<br>2 1 3 | 5 2<br>2 3 1 2 4 | 3 2<br>999999 999999 999999 |
| **output** | **output** | **output** |
| 2 | 43 | 990979013 |

# Problem F - Frogs

Time limit: 0.3 s
Memory limit: 512 MiB

An infinite sequence of lily pads is floating on a lake, neatly arranged in a straight line and numbered with integers starting from 1 and increasing from left to right. Each morning, an army of frogs arrives and starts jumping on the lily pads according to the following rules:

- There are $n$ types of frogs denoted with integers 1 through $n$. There are a total of $t_k$ frogs of type $k$ arriving.
- The frogs arrive on lilypads one by one ordered by their type increasingly.
- A frog of type $k$ jumps on the lily pads $k$, $2k$, $3k$, etc. until it reaches an empty lily pad. Then it sits on that lily pad and leisurely croaks for the rest of the day.

For each type $k$, find the highest-numbered lily pad reached by a frog of that type.

## Input

The first line contains an integer $n$ ($1 \leq n \leq 10$) – the number different types of frogs. The $k$-th of the following $n$ lines contains an integer $t_k$ ($1 \leq t_k \leq 50\,000\,000$) – the number of frogs of type $k$.

## Output

Output $n$ lines. The $k$-th line should contain a single integer – the highest-numbered lily pad reached by a frog of type $k$.

## Example

**input**

```
3
2
2
2
```

**output**

```
2
6
9
```

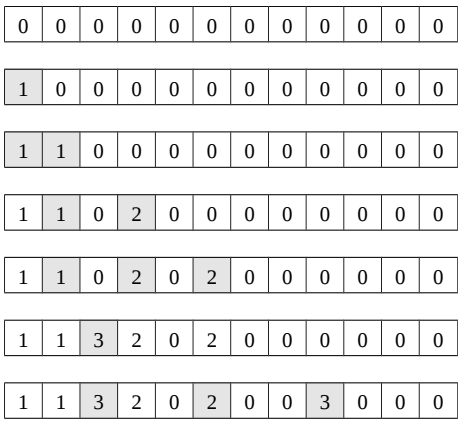
**input**

```
3
10000000
10000000
10000000
```

**output**

```
10000000
30000000
50000001
```

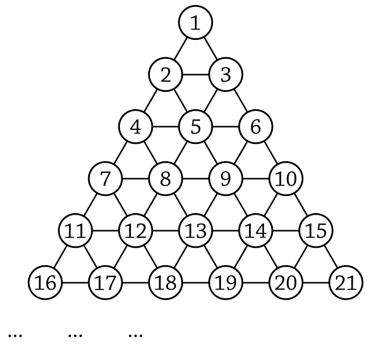
The first example input is depicted below. The number denotes the frog type resting at the lily pad after each frog arrives (zero denotes an empty lily pad). The shaded lily pads are those that the current frog jumped on.



# Problem G - Grid

Time limit: 0.3 s
Memory limit: 512 MiB

A triangular grid is constructed by organizing all positive integers in rows top to bottom where the $k$-th row contains $k$ successive integers from left to right. The rows are center aligned and each integer is connected with a unit-sized segment to its neighbours in the same row as well as the neighbouring elements above and below, as in the figure below.




A *regular triangle* is a tuple of integers $(a, b, c)$ such that the triangle formed by vertices $a, b, c$ is an equilateral triangle whose sides are parallel to the sides of the triangle $1, 2, 3$. For example, $(4, 6, 13)$ is a regular triangle, while $(2, 6, 8)$ is not.

Given two integers $a$ and $b$, find all integers $c$ which form a regular triangle with $a$ and $b$.

## Input

The first line contains two different integers $a$ and $b$ ($1 \leq a, b \leq 500\,000\,000$).

## Output

If there are no integers $c$ that form a regular triangle with $a$ and $b$, output `none`. Otherwise, output all possible integers $c$ on a single line, sorted in the increasing order.

## Example

| input | input | input |
|---|---|---|
| 2 6 | 2 9 | 6 4 |
| **output** | **output** | **output** |
| none | 7 | 1 13 |

# Problem H - Happy

Time limit: 2 s
Memory limit: 512 MiB

We say that an integer sequence of size $n$ is *happy* if the absolute differences between consecutive elements cover all possible values between 1 and $n-1$ inclusive. For example, the sequence $(1, 3, 4, 2)$ is not happy since the absolute differences between consecutive elements are $(2, 1, 2)$ – the number 3 is missing. On the other hand, the sequence $(3, 1, 4, 3)$ is happy.

You are given an unhappy sequence of integers $a$. Find all happy sequences $b$ that can be obtained from $a$ by *replacing exactly one element* with an arbitrary integer.

## Input

The first line contains an integer $n$ ($2 \leq n \leq 1\,000\,000$) – the number of elements in sequence $a$. The following line contains $n$ integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_k \leq 1\,000\,000$) – the sequence $a$. The sequence $a$ will not be happy.

## Output

The first line should contain a single integer $m$ – the number of happy sequences $b$. The following $m$ lines should contain instructions on how to obtain the sequences from $a$ (in arbitrary order).

Each of those lines should contain two integers $j$ and $x$ – the index of the element in the sequence $a$ that should be changed and the new value at that index. The index $j$ should be an integer between 1 and $n$ inclusive while $x$ can be an arbitrary signed 32-bit integer.

## Example

**input**

```
4
1 3 5 3
```

**output**

```
2
2 2
2 4
```

**input**

```
3
1 1 2
```

**output**

```
4
1 -1
1 3
2 0
2 3
```

# Problem I - Inquiry

Time limit: 2 s
Memory limit: 512 MiB

We are given two strings $a$ and $b$, and are interested in permutations of $a$ that contain $b$ as a substring.

Find the number of such permutations modulo 10 007.

## Input

The first line contains the string $a$. The second line contains the string $b$. Both strings consists of lowercase letters from the English alphabet. It holds $1 \leq |b| \leq |a| \leq 500$, where $|s|$ denotes the length of the string $s$.

## Output

Output a single integer – the number of permutations modulo 10 007.

## Example

| input | input | input |
|---|---|---|
| mirko<br>mir | sssss<br>ss | barbarabarbara<br>barbabarba |
| **output** | **output** | **output** |
| 6 | 1 | 30 |

In the first example, the six permutations that contain the substring "mir" are "kmiro", "komir", "mirko", "mirok", "okmir" i "omirk".

# Problem J - Jolt

Time limit: 1 s
Memory limit: 512 MiB

Luka is building a simple mobile game. The game board is a table consisting of 3 rows and $n$ columns. Initially, the first row contains a *permutation* of integers 1 through $n$, while the second and the third row contain arbitrary integers between 1 and $n$ inclusive.

In each step, a player chooses a column and removes it from the game board with an abrupt jolt. After the column flies out, the remaining columns on the right slide leftwards until they lock with the rest of the columns and form a new table.

After some number of steps the player pushes a button and the contents of each row is sorted in the increasing order. The player wins if, after sorting, all three rows of the table are identical.

Given an initial game board, find the minimal number of steps needed for the player to win.

## Input

The first line contains an integer $n$ ($1 \leq n \leq 100\,000$) – the number of columns in the initial game board. Each of the three following lines contains a sequence of $n$ integers between 1 and $n$ inclusive – one row of the game board. There will be no duplicates in the first row of the board.

## Output

The first line should contain a single integer – the minimal number of steps needed for the player to win.

## Example

**input**

```
7
5 4 3 2 1 6 7
5 5 1 1 3 4 7
3 7 1 4 5 6 2
```

**output**

```
4
```

**input**

```
9
1 3 5 9 8 6 2 4 7
2 1 5 6 4 9 3 4 7
3 5 1 9 8 6 2 8 7
```

**output**

```
2
```

In the first example, the game can be won by erasing columns $2, 4, 6$ and 7. After sorting, the contents of each row is the sequence $(1, 3, 5)$. It is impossible to win the game with fewer than four steps.

# Problem K - Key

Time limit: 2 s
Memory limit: 512 MiB

Goran is developing a toy cryptosystem based on the *multi-prime RSA* algorithm. An important part of every key in this cryptosystem is *modulus* - a product of *two or more distinct prime numbers*. The security of the system depends (among other things) on the infeasibility of factoring the modulus.

Goran's implementation of the key generation algorithm has multiple flaws. Firstly, there is not enough entropy in the random number generator and only 36 different fixed primes are used for key generation. Therefore, each modulus is a product of some subset of those primes. Furthermore, all of those primes are less than $10^9$.

Goran has generated $n$ different moduli and has asked his friend Ivan to help him test the system. Ivan (who knows the flaws of the implementation) wants to prove to Goran he can completely factor all the moduli and also communicate the *proof of factorization* in the most efficient way.

The proof of factorization for a modulus $m$ is a set of prime numbers $S$ with the property that, when $m$ is divided with all those primes in $S$ which divide $m$, the result is either 1 or a prime number. The proof of factorization for a set of moduli $M$ is a set of prime numbers $S$ which is a proof of factorization for every $m$ from $M$.

Given n moduli, find the size (the number of elements) of the smallest proof of factorization.

## Input

The first line contains an integer $n$ ($1 \leq n \leq 100\,000$) – the number of moduli. The $k$-th of the following $n$ lines contains the $k$-th modulus $x_k$ ($2 \leq x_k \leq 10^{18}$).

Evey modulus $x_k$ is a product of two or more prime number smaller than $10^9$. The total number of different primes appearing as factors across all moduli is at most 36.

## Output

Output a single integer – the size of the smallest proof of factorization.

## Example

**input**

```
2
35
77
```

**output**

```
1
```

**input**

```
5
6
35
77
14
21
```

**output**

```
2
```

**input**

```
1
999992936000441063
```

**output**

```
1
```

In the first example, the set $\{7\}$ is a minimal proof of factorization – when we divide the moduli with 7 the results are 5 and 11 which are both prime.

In the second example, both $\{7,2\}$ and $\{7,3\}$ are minimal proofs of factorization.

In the third example, the modulus is equal to $999992999 * 999999937$ and either of those primes is the minimal proof of factorization.

# Problem L - League

Time limit: 1.5 s
Memory limit: 512 MiB

A railroad network of a certain country consists of $n$ cities numbered 1 through $n$ and connected with $n-1$ direct tracks. There is a unique path between every pair of cities and, hence, the network forms a tree. The cost of a single ticket for travel between two cities is equal to the total number of direct tracks on the path connecting the cities.

A competitive programming league has members throughout the country and, each year, organizes a programming contest that all members attend. The contest is hosted in one of the cities and all the members travel there by train. The *cost* of the contest is the total cost of all the tickets for the individual members.

In the first year, the contest is organized in city 1. Every following year the location of the contest is moved to a neighboring city. In other words, the city where the contest is held in year $k$ is always connected with a direct track to the city where the contest was held year in year $k-1$.

You are given the initial number of league members in each of the cities. You are also given a list of $m$ events in chronological order, where each event is one of the following:

- `P` $x$ - a new member joins the league in city $x$.
- `S` $x$ - a contest is being held in city $x$.

Find the cost of each contest held.

## Input

The first line contains an integer $n$ ($1 \leq n \leq 100\,000$) – the number of cities. The following line contains $n$ integers $h_1, h_2, \ldots, h_n$ ($0 \leq h_k \leq 1000$) – $h_k$ is the initial number of members city $k$.

Each of following $n-1$ lines contains two different integers $a$ and $b$ ($1 \leq a, b \leq n$) describing a direct railroad track between cities $a$ and $b$. The railroad network forms a tree.

The following line contains an integer $m$ ($1 \leq m \leq 300\,000$) – the number of events. Each of the following $m$ lines contains one event in the form '$c$ $x$', where $c$ is either `P` or `S` and $x$ an integer between 1 and $n$ inclusive – the type of the event and the city, respectively. The events are given in chronological order.

The first event will always be `S 1`. For two subsequent events of type `S`, possibly with many events of type `P` in between, the cities in those `S` events will be different and directly connected with a railroad track.

## Output

For each event of type `S`, in chronological order, output a line containing a single integer – the cost of the corresponding contest.

## Example

**input**

```
5
1 1 0 1 3
1 5
3 5
4 5
2 1
6
S 1
S 5
S 3
P 3
P 4
S 5
```

**output**

```
6
4
10
6
```

**input**

```
3
2 1 4
1 2
2 3
6
S 1
P 1
P 2
S 2
P 1
S 1
```

**output**

```
9
7
10
```