Problem A
ABC String
Time Limit: 1 second

You're given a string consisting of the characters `A`, `B`, and `C`. The string contains the same count of `A`, `B`, and `C` characters.

A string is *beautiful* if

- Its length is divisible by 3.
- The string can be split evenly into contiguous substrings of size 3, where each substring has one `A`, one `B`, and one `C`, in any order.

For example: `ABCCBA` is a beautiful string, but `ABCAB` and `CCBAAB` are not beautiful.

Given a string, you want to partition it into subsequences (not necessarily contiguous) such that each subsequence is a beautiful string.

For example, for the string `ABACBCAACCBB`, we can do the following:

```
AB    CA C B
  ACB   A C B
```

This partitions the string into two subsequences `ABCACB` and `ACBACB`, both of which are beautiful strings.

For the given string, find the minimum number of subsequences you can partition it into such that each subsequence is beautiful. It can be proven that there is always at least one such partition for all possible inputs that satisfy the input constraints.

Input

The first line of input contains a string s ($3 \leq |s| \leq 3 \cdot 10^5$). $|s|$ is divisible by 3. s contains an equal number of characters `A`, `B`, and `C`.

Output

Output a single integer, which is the minimum subsequences that s can be partitioned into so each subsequence is a beautiful string.

Sample Input 1	Sample Output 1
ABACBCAACCBB	2

Problem B
Balanced Tree Path
Time Limit: 2 seconds

You are given a tree where each node is annotated with a character from `()[]{}`. A path is a sequence of one or more nodes where no node is repeated and every pair of adjacent nodes is connected with an edge. A path is balanced if the characters at each node, when concatenated, form a balanced string. A string is balanced if it satisfies the following definition:

- An empty string is balanced.
- If s is a balanced string, then `(`s`)`, `[`s`]`, and `{`s`}` are balanced strings.
- if a and b are balanced strings, then ab (a concatenated with b) is a balanced string.

Compute the number of balanced paths over the entire tree.

Input

The first line of input contains a single integer n ($2 \leq n \leq 5 \cdot 10^3$).

The next line contains a string of n characters, where each character is one of `()[]{}`.

Each of the next $n-1$ lines contains two integers, u and v ($1 \leq u < v \leq n$), indicating that nodes u and v are connected with an edge. It is guaranteed the graph is a tree.

Output

Output a single integer, which is the number of balanced paths over the entire tree.

Sample Input 1	Sample Output 1
4 ()() 1 2 2 3 3 4	4

Sample Input 2

```
4
[[]]
1 2
2 3
3 4
```

Sample Output 2

```
2
```

Sample Input 3

```
6
([]{})
1 2
2 3
3 4
4 5
5 6
```

Sample Output 3

```
4
```

Problem C
A Cappella Recording

Time Limit: 1 second

Geoffry is preparing an a cappella composition where he sings the entire song by himself.

Each note of the song has a pitch between 0 and 10^9. Because of the varying pitches in the song, Geoffry will record himself singing multiple times. In a single recording, he will pick some subset of the notes to sing and he will sing exactly those notes. To avoid straining his voice too much, within a single recording, there is a limit to the difference between the maximum pitch and the minimum pitch among the notes he sings.

Compute the minimum number of times that Geoffry can record himself singing the song and each note is sung in at least one of the recordings.

Input

The first line contains two integers n and d ($1 \leq n \leq 10^5, 0 \leq d \leq 10^9$), where n is the number of notes in Geoffry's song, and d is the largest difference between the minimum pitch and the maximum pitch that Geoffry can handle.

Each of the next n lines contains a single integer p ($0 \leq p \leq 10^9$). These are the pitches of the notes in Geoffry's song, in the order that they are to be sung.

Output

Output a single integer, which is the minimum number of times that Geoffry can record himself singing the song and each note is sung in at least one of the recordings.

Sample Input 1	Sample Output 1
6 0 3 1 4 1 5 9	5

Sample Input 2

```
6 1
3
1
4
1
5
9
```

Sample Output 2

```
4
```

Sample Input 3

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

Sample Output 3

```
3
```

Sample Input 4

```
6 4
3
1
4
1
5
9
```

Sample Output 4

```
2
```

Sample Input 5

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

Sample Output 5

```
1
```

Problem D
Ordered Problem Set
Time Limit: 1 second

You are running a programming contest that features n problems of distinct difficulties. You wish to announce ahead of time that the problems are ordered in such a way that, if the problems are divided into k sections numbered 1 through k, each with exactly $\frac{n}{k}$ problems, and problem p is assigned to section $\lceil \frac{kp}{n} \rceil$, then for every pair of sections i and j with $i < j$, every problem in section i is easier than every problem in section j. Note that k must be greater than 1 and be a factor of n.

However, you have just sent your problems to the printer so the order cannot be changed. For what values of k would this claim be true?

Input

The first line of input contains a single integer n ($2 \leq n \leq 50$), which is the number of problems.

Each of the next n lines contains a single integer d ($1 \leq d \leq n$). These are the difficulties for the problems in the order that they appear in the problem set. The difficulties are distinct. The problem with difficulty 1 is the easiest problem and the problem with difficulty n is the hardest problem.

Output

Output a list of integers, one per line. The integers are all valid values of k in increasing order. If no such values exist, output -1.

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

Sample Input 2

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

Sample Output 2

```
2
3
6
```

Sample Input 3

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

Sample Output 3

```
-1
```

Problem E
Acceptable Seating Arrangements
Time Limit: 1 second

Charlie is managing a classroom. The seats in the classroom are arranged in a grid with rows and columns. Each student has a distinct height.

A configuration of students to seats is *acceptable* if the following conditions are met:

- Each student is assigned to exactly one seat.
- The students are seated in increasing order of height from left to right in each row.

The students are initially seated in an acceptable arrangement. Charlie wants to rearrange students into a potentially different acceptable arrangement. To do this, he can swap any two students. However, he wants to ensure that the configuration stays acceptable after each swap.

Help Charlie devise a strategy to move the students from the original arrangement to his preferred arrangement. You don't need to minimize the number of swaps, but you are limited to at most 10^4 swaps.

It can be proven that this is always possible for all possible inputs that satisfy the input constraints.

Input

The first line of input contains two integers r and c ($1 \leq r, c \leq 20$). Charlie's classroom has r rows and c columns of seats.

Each of the next r lines contains c integers h ($1 \leq h \leq r \cdot c$), representing the heights of the students in each row in the original arrangement. The heights are guaranteed to be distinct, and the arrangement is guaranteed to be acceptable.

Each of the next r lines contains c integers h ($1 \leq h \leq r \cdot c$), representing the heights of the students in each row in Charlie's desired arrangement. The heights are guaranteed to be distinct, and the arrangement is guaranteed to be acceptable.

Output

On the first line, output an integer k, which is the number of swaps to perform ($0 \leq k \leq 10^4$).

Then, output the k swaps which change the original arrangement to Charlie's preferred arrangement. On each of the next k subsequent lines, output four integers r_1, c_1, r_2, c_2 ($1 \leq r_1, r_2 \leq r$, $1 \leq c_1, c_2 \leq c$, $(r_1, c_1) \neq (r_2, c_2)$). This represents a swap of the student in row r_1, column c_1 with the student in row r_2, column c_2.

It can be proven that it's always possible to accomplish this in under 10^4 swaps for all possible inputs that satisfy the input constraints. Remember that the arrangement must be acceptable after each swap.

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

Problem F
Four Die Rolls
Time Limit: 1 second

Ashley and Brandon are playing a game where they roll a six-sided dice four times. Each integer from 1 to 6 is on one face of the dice and all values are equally likely to appear. Each time they roll the dice, they write down the value that shows on the top face of the dice, eventually forming a four-digit integer.

Ashley wins if all four digits in the integers are different. Otherwise, Brandon wins.

The dice has been rolled from 1 to 3 times so far. Compute the number of different ways to roll the remaining times such that Ashley wins, and the number of different ways to roll the remaining dice such that Brandon wins. Two ways are different if the integers formed are different - for example, rolling a 1, then a 2, then a 3, then a 4 forms the integer 1234, and is different from rolling a 1, then a 2, then a 4, then a 3, which would form the integer 1243.

Input

The first line contains n ($1 \leq n \leq 3$), which is the number of times the dice have been rolled so far.

The next line contains n integers ranging from 1 to 6, which are the first n rolls of the dice in order.

Output

Output two space-separated integers on one line, which are the number of ways to roll the remaining times such that Ashley wins, and the number of ways to roll the remaining dice such that Brandon wins.

Sample Input 1	Sample Output 1
2 1 2	12 24

Sample Input 2	Sample Output 2
2 3 3	0 36

This page is intentionally left blank.

Problem G
Rampant Growth
Time Limit: 1 second

Eddy is planning his garden, which can be represented as a grid. He wants his garden to have exactly one plant in each column. To make sure that plants do not compete for resources, if two plants are in adjacent columns, they must be in different rows.

Compute the number of different ways he can place plants in his garden to respect the above conditions. Two ways are different if one square has a plant in one arrangement but does not have a plant in the other. Because the number of ways may be large, output the number of ways modulo $998\,244\,353$.

Input

The single line of input contains two space-separated integers r and c $(1 \leq r, c \leq 50)$, where r is the number of rows in Eddy's garden grid, and c is the number of columns.

Output

Output a single integer, which is the number of ways Eddy can place plants in his garden, modulo $998\,244\,353$.

Sample Input 1	Sample Output 1
3 2	6

Sample Input 2	Sample Output 2
1 5	0

Sample Input 3	Sample Output 3
42 25	722210361

This page is intentionally left blank.

Problem H
Triple Sevens

Time Limit: 1 second

Eddy is overseeing construction of some new slot machines. His slot machines consist of three wheels, each of which can show one of several different digits. When activated, each wheel shows a random digit. The goal is for all wheels to be able to show the digit 7.

A slot machine is good if each wheel is capable of showing the digit 7, and bad otherwise. For a given slot machine, determine if it is good or bad.

Input

The first line contains a single integer n ($1 \leq n \leq 10$), which is the number of different digits on each wheel. Each wheel has the same number of digits.

Each of the next three lines contains n distinct digits (in the range from 0 to 9). These are the digits on each of the three wheels.

Output

Output a single integer, which is 777 if the slot machine is good, 0 otherwise.

Sample Input 1	Sample Output 1
2 0 7 7 3 7 0	777

Sample Input 2	Sample Output 2
2 0 7 7 3 3 9	0

This page is intentionally left blank.

Problem I
Item Selection
Time Limit: 1 second

You are browsing a website that lists items for sale. The website has a paging UI that displays a fixed number of items per page, one page at a time.

For example, if there are 55 items and the page displays exactly 20 at a time, then there are 3 pages in total. Items 1 through 20 are on page 1, items 21 through 40 are on page 2, and items 41 through 55 are on page 3.

You may navigate and select items using these UI elements:

- A checkbox for every item on the current page. After you click a checkbox, a selected item becomes unselected, and an unselected item becomes selected. You cannot click a checkbox for an item that is not on the current page.
- A "Select All" button. All unselected items on the current page become selected after you click this button.
- A "Deselect All" button. All selected items on the current page become unselected after you click this button.
- A "Next Page" button. Clicking it navigates to the next page and increments the current page number by one. This button is disabled on the last page.
- A "Previous Page" button. Clicking it navigates to the previous page and decrements the current page number by one. This button is disabled on the first page.

The website has pre-selected some items for you based on its machine learning recommendation algorithm. The recommendation may or may not work for you. You know exactly the items that you want to purchase, which may differ from the pre-selected items. What is the minimum number of checkbox and button clicks required to select exactly the items you actually want?

Input

The first line of input has five integers n, m ($1 \leq m \leq n \leq 10^3$), s ($1 \leq s \leq \lceil \frac{n}{m} \rceil$), p, q ($0 \leq p, q \leq n$), where:

- n is the number of items. The items have item numbers from 1 to n.
- m is the fixed number of items displayed per page.
- s is the number of the page currently displayed.
- p is the number of preselected items.
- q is the number of items you want.

Each of the next p lines contains an integer i ($1 \leq i \leq n$). These are the item numbers of the preselected items. These p items are distinct and are listed in increasing order. It is possible that the website has pre-selected none of the items ($p = 0$), in which case the input has no lines for pre-selected items.

Each of the next q lines contains an integer j ($1 \leq j \leq n$). These are the item numbers of the items you want to buy. These q items are distinct and are listed in increasing order. It is possible that you want to buy none of the items ($q = 0$), in which case the input has no lines for items you want.

Output

Output a single integer, which is the minimum number of checkbox and button clicks required to select exactly the items you want.

Sample Input 1	Sample Output 1
11 4 1 5 5 1 4 9 10 11 1 3 6 7 8	7

Problem J
Sequence Guessing
Time Limit: 4 seconds

You are tasked with creating a secret sequence of integers that is difficult to guess.

The sequence is subject to the following constraints:

- The first number in the sequence must be 0.
- The last number in the sequence must be $100\,000$.
- Each number in the sequence must be either 1 or 2 greater than the one preceding it.

At first, all you need to reveal is the length of the sequence.

Then, an adversary will guess the numbers in the sequence one at a time.

- If the guessed number is in your sequence, you must reveal exactly where in the sequence it appears.
- If the guessed number is not in your sequence, you must simply reveal that it is not in the sequence. This is considered a "miss".

Note that because you are not forced to write down the sequence in advance, you can "cheat" by changing the sequence you have in mind, so long as it does not contradict the information you have revealed so far. It turns out that under these conditions, you can always force the adversary to get $33\,333$ misses before they can guess every number in your sequence. Your job is to write a program that does so.

Interaction

This is an interactive problem.

Your program should begin by printing k ($2 \leq k \leq 100\,001$), the length of your sequence, as a single integer on a single line. After this, you will receive one integer x on a single line. This integer is guaranteed to be between -1 and $100\,000$ inclusive.

- If $x = -1$, the adversary has given up; your program should print all k integers in your sequence in order, one line per integer, and then exit. The adversary is guaranteed to give this input after it has gotten $33\,333$ misses, though it may do so earlier. After printing all

k integers, your program should exit. If you print a valid sequence consistent with your previous responses, your submission will be considered correct for this test case.

- If x is not in your sequence, print -1 on a single line.
- If x is in your sequence, print a single integer i on a single line, such that x is the ith (1-indexed) number in the sequence. If you have printed every integer from 1 to k, your program should now exit, and your submission will receive a Wrong Answer verdict.

Do not forget to flush the output after every integer you print.

After this, if your program has not yet exited, the process will repeat, with you receiving another single integer. The adversary is guaranteed to never repeat integers.

The adversary may employ different guessing strategies on different runs.

Read	Sample Interaction 1	Write
	50001	
0		
	1	
1		
	-1	
-1		
	0 2 <omitted 49997 lines for brevity> 99998 100000	

Problem K
Streets Behind
Time Limit: 1 second

Your running club has some serious runners and some casual runners. You schedule several training runs with a mixture of serious runners and casual runners. Serious runners run at a faster pace than casual runners, and will leave them behind.

You want all the runners to become serious runners, so when you schedule training runs, you carefully choose the number of serious and casual runners who will participate. You know that when there are x serious runners and y casual runners in a training run, if $\frac{x}{x+y}$ is greater than or equal to a threshold $\frac{a}{b}$, then after the run, all y casual runners, feeling the pressure to keep up with the serious runners become serious runners moving forward.

Compute the minimum number of training runs you need to convert all members of the club into serious runners.

Input

The first line of input contains a single integer t ($1 \leq t \leq 100$). This is the number of test cases.

Each of the next t lines contains a test case. Each test case consists of four integers n, k, a, b ($1 \leq n, k, a, b \leq 10^9$ and $a \leq b$), where n is the number of serious runners, k is the number of casual runners, and $\frac{a}{b}$ is the threshold which converts casual runners to serious runners.

Output

Output t lines, each containing a single integer. For each test case, output the minimum number of training runs needed to convert all casual runners to serious runners. If it is impossible to convert all casual runners to serious runners, output -1 for that test case.

Sample Input 1	Sample Output 1
3 9 5 5 6 2 7 1 8 3 4 1 5	3 1 1

This page is intentionally left blank.

Problem L
Training

Time Limit: 1 second

Ashley is training for a programming contest on Brandon's Online Judge. Brandon's Online Judge has a new feature which allows Ashley's coach, Tom, to load a list of problems for Ashley.

Tom has curated some problems for Ashley to work on. Each problem has two integers as a lower skill bound and an upper skill bound. Each programmer has an integer skill level. If someone with a skill level between the lower and upper bounds of a problem (inclusive), and they solve that problem, then his/her skill level goes up by 1.

Ashley will train on Tom's curated list of problems as follows – she will look at the first problem on the list and either solve it or skip it. She will repeat this for every problem on the list in the order Tom loaded the problems. Once she has skipped a problem, she can never go back to it.

Compute the maximum skill level Ashley can have if she chooses to solve or skip problems optimally.

Input

The first line contains two integers n and s ($1 \leq n \leq 10^5, 0 \leq s \leq 10^9$), where n is the number of problems Tom has curated for Ashley, and s is Ashley's current skill level.

Each of the next n lines contains two integers l and r ($0 \leq l \leq r \leq 2 \cdot 10^9$). These are the lower ($l$) and upper ($r$) skill bounds on each of Tom's problems, in the order that Tom loaded them.

Output

Output a single integer, which is the maximum skill level Ashley can attain.

Sample Input 1	Sample Output 1
3 0 0 2 0 0 1 1	2

This page is intentionally left blank.

Problem M
Champernowne Verification

Time Limit: 1 second

The k^{th} Champernowne word is obtained by writing down the first k positive integers and concatenating them together. For example, the 10^{th} Champernowne word is 12345678910.

Given a positive integer n, determine if it is a Champernowne word, and if so, which word.

Input

The first line contains a single integer, n ($1 \leq n \leq 10^9$). n will not have leading zeroes.

Output

If n is the k^{th} Champernowne word, output k. Otherwise, output -1.

Sample Input 1	Sample Output 1
123456789	9

Sample Input 2	Sample Output 2
1000000000	-1

Sample Input 3	Sample Output 3
11	-1

Sample Input 4	Sample Output 4
1324	-1

This page is intentionally left blank.