Q:-1

Hussain is very bored, and has a lot of exams lined up next week. But he doesn't want to study. As usual, he decided to fail his exams and play with Hasan instead (who has already failed). Hussain invented this new game to play with Hasan.

Hussain shows Hasan a multiset of integers. (A multiset is a collection of elements where there can be duplicates). In every move Hussain removes a maximum number from this multiset and divides it by 2 (integer division, which is rounded down).

If after dividing, the number is still positive (greater than 0) he re-inserts it back into the multiset.

Before the start of the game, Hussain shows Hasan the numbers in his multiset.

Hussain asks Hasan, **M** questions. The **i-th** question will be denoted by **Q[i]**, and Hasan must find the value of the number Hussain will be dividing (before he performs the division) after **Q[i]**-1 moves. That is, what is the value of the number on which the **Q[i]**-th division is performed?

Can you help Hasan and answer the queries?

**Input**

* The first line contains 2 space separated integers **N, M** denoting the initial size of the multiset and the number of questions respectively.
* The next line contains **N** space separated integers, the initial elements of the multiset.
* Of the next **M** lines, the **i-th** line will contain **Q[i]**.

**Output**

* Output **M** lines, the i-th of which contains the answer for the i-th query.

**Constraints**

* 1 ≤ **N, M** ≤ 106
* The queries are given in chronological order. That is, **Q[i] > Q[i-1]**
* Every element of the multiset will be a positive integer less than 263
* It's guaranteed that the set will contain at least one element at the time of any query.

**Example**

**Input:**

4 6

8 5 3 1

1

2

3

4

6

8

**Output:**

8

5

4

3

2

1

**Explanation**

We show the multiset sorted at each step, for convenience.

* Before any move, the multiset is (8, 5, 3, 1).
* In the first move, 8 is removed, dived by 2, and thus becomes 4, and then re-inserted. So, the multiset, after the first move is (5, 4, 3, 1).
* In the second move, 5 is removed and after division, it become 2, which is re-inserted. The multiset becomes (4, 3, 2, 1).
* After the third move, the multiset becomes (3, 2, 2, 1).
* After the fourth move, it becomes (2, 2, 1, 1).
* After the fifth move, it becomes (2, 1, 1, 1).
* After the sixth move, it becomes (1, 1, 1, 1).
* In the seventh move, 1 is removed, and on division, it no longer is greater than 0. Hence, it is not re-inserted. So, after the seventh move, the multiset becomes (1, 1, 1).
* After the eight move, the multiset becomes (1, 1).

The value being divided on the first move is 8. Hence the first output is 8.

The value being divided on the first move is 5. Hence the first output is 5.

The value being divided on the eight move is 1. Hence the last output is 1.

Q-2:

As you might remember, the collector of Siruseri had ordered a complete revision of the Voters List. He knew that constructing the list of voters is a difficult task, prone to errors. Some voters may have been away on vacation, others may have moved during the enrollment and so on.

To be as accurate as possible, he entrusted the task to three different officials. Each of them was to independently record the list of voters and send it to the collector. In Siruseri, every one has a ID number and the list would only list the ID numbers of the voters and not their names. The officials were expected to arrange the ID numbers in ascending order in their lists.

On receiving the lists, the Collector realised that there were discrepancies - the three lists were not identical. He decided to go with the majority. That is, he decided to construct the final list including only those ID numbers that appeared in at least 2 out of the 3 lists. For example if the three lists were

23 30 42 57 90

21 23 35 57 90 92

21 23 30 57 90

then the final list compiled by the collector would be:

21 23 30 57 90

The ID numbers 35, 42 and 92 which appeared in only one list each do not figure in the final list.

Your task is to help the collector by writing a program that produces the final list from the three given lists.

**Input format**

The first line of the input contains 3 integers *N*1, *N*2 and *N*3. *N*1 is the number of voters in the first list, *N*2 is the number of voters in the second list and *N*3 is the number of voters in the third list. The next *N*1 lines (lines 2,...,*N*1+1) contain one positive integer each and describe the first list in ascending order. The following *N*2lines (lines *N*1+2,...,*N*1+*N*2+1) describe the second list in ascending order and the final *N*3 lines (lines *N*1+*N*2+2,...,*N*1+*N*2+*N*3+1) describe the third list in ascending order.

**Output format**

The first line of the output should contain a single integer *M* indicating the number voters in the final list. The next *M* lines (lines 2,...,*M*+1) should contain one positive integer each, describing the list of voters in the final list, in ascending order.

**Test data**

You may assume that 1 ≤ *N*1,*N*2,*N*3 ≤ 50000.

**Example**

**Sample input:**

5 6 5

23

30

42

57

90

21

23

35

57

90

92

21

23

30

57

90

**Sample output:**

5

21

23

30

57

90

Q-3:

You are given a string **s** with length **n**. You should find a [permutation](https://en.wikipedia.org/wiki/Permutation) **P** of numbers 1 through **n** such that if you apply this permutation on the string **s**, you will get a [palindromic](https://en.wikipedia.org/wiki/Palindrome) string.

The result of applying a permutation **P** on the string **s** is a string **t** with length **n**such that for each **i** (1 ≤ **i** ≤ **n**), the **i**-th character of **t** is given as as **t**[**i**] = **s**[**Pi**].

**Input**

* The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows.
* The first and only line of each test case contains the string **s**.

**Output**

For each test case, print a single line. If it is impossible to find a valid permutation **P**, this line should contain a single integer -1. Otherwise, it should contain **n** space-separated integers **P1, P2, ..., Pn**.

If there are multiple valid permutations, you may print any one.

**Constraints**

* 1 ≤ **n** ≤ 105
* **s** will consist only of lowercase English letters (i.e. characters 'a' through 'z')

**Subtasks**

**Subtask #1 (20 points)**: 1 ≤ **T**, **n** ≤ 10

**Subtask #2 (20 points)**: 1 ≤ **T**, **n** ≤ 100

**Subtask #3 (60 points)**: 1 ≤ **T** ≤ 10

**Example**

**Input**

4

aa

baa

abc

abab

**Output**

1 2

2 1 3

-1

1 2 4 3

**Explanation**

**Example case 1:** The string **t** obtained using the identity permutation will have **t**[1] = **s**[1] and **t**[2] = **s**[2]. That means **t** = "aa", which is a palindrome.

**Example case 2:** The characters of the string **t** obtained by applying the permutation **2, 1, 3** are **t**[1] = **s**[2], **t**[2] = **s**[1] and **t**[3] = **s**[3]. Therefore, **t** = "aba", which is a palindrome.

**Example case 3:** There is no way to find a permutation **P** such that we can obtain a palindrome from **s** using it.

**Example case 4:** Applying the permutation **1, 2, 4, 3** on **s** results in **t** = "abba", which is a palindrome. Another permutation that you may apply is **2, 1, 3, 4**; this results in **t** = "baab", which is also a palindrome.

Q:-4

**Love for Characters**

Send Feedback

#### Ayush loves the characters ‘a’, ‘s’, and ‘p’. He got a string of lowercase letters and he wants to find out how many times characters ‘a’, ‘s’, and ‘p’ occurs in the string respectively. Help him find it out.

##### Input:

First line contains an integer denoting length of the string.

Next line contains the string.

##### Constraints:

1<=n<=10^5

‘a’<= each character of string <= ‘z’

##### Output:

Three space separated integers denoting the occurrence of letters ‘a’, ‘s’ and ‘p’ respectively.

##### Sample Input:

6

aabsas

##### Sample output:

3 2 0

Q:-5

**Different Names**

Send Feedback

#### In Little Flowers Public School, there are many students with same first names. You are given a task to find the students with same names. You will be given a string comprising of all the names of students and you have to tell the name and count of those students having same. If all the names are unique, print -1 instead.

#### Note: We don't have to mention names whose frequency is 1.

##### Input Format:

The only line of input will have a string ‘str’ with space separated first names of students.

##### Output Format:

Print the names of students along with their count if they are repeating. If no name is repeating, print -1

##### Constraints:

1 <= |str| <= 10^5

Time Limit: 1 second

##### Sample Input 1:

Abhishek harshit Ayush harshit Ayush Iti Deepak Ayush Iti

##### Sample Output 1:

harshit 2

Ayush 3

Iti 2

##### Sample Input 2:

Abhishek Harshit Ayush Iti

##### Sample Output:

-1

Q:-6

**Extract Unique characters**

Send Feedback

#### Given a string, you need to remove all the duplicates. That means, the output string should contain each character only once. The respective order of characters should remain same.

#### Input format :

String S

##### Output format :

Output String

##### Constraints :

1 <= Length of S <= 50000

##### Sample Input 1 :

ababacd

##### Sample Output 1 :

abcd

##### Sample Input 2 :

abcde

##### Sample Output 2 :

abcde

Q:-7

**Warm Reception**

Send Feedback

#### There is only one beauty parlour in the town CodingNinjasLand. The receptionist at beauty parlour is flooded with appointment requests because “Hakori” festival is round the corner and everyone wants to look good on it.

#### S/he needs your help. The problem is they don’t have chairs in reception. They are ordering chairs from NinjasKart. They don’t want to order more than required. You have to tell minimum number of chairs required such that none of the customer has to stand.

##### Input Format :

First line contains number of customers that will come. Second line contains N space separated integers which represent arrival timings of the customer. Third line contains N space separated integers which represent departure timings of the customer. Arrival and departure timings are given in 24 hour clock.

##### Constraints:

1<= N <= 100

Arrival and departure timings lie in the range [0000 to 2359]

Time Limit: 1 second

##### Output Format :

You have to print the minimum number of chairs required such that no customer has to wait standing.

##### Sample Test Cases:

#### Sample Input 1 :

5

900 1000 1100 1030 1600

1900 1300 1130 1130 1800

#### Sample Output 1:

4

#### Explanation:

4 because at 1100 hours, we will have maximum number of customers at the shop, throughout the day. And that maximum number is 4.

Q8:-

**Tell the positions**

Send Feedback

#### In a class there are ‘n’ number of students. They have three different subjects: Data Structures, Algorithm Design & Analysis and Operating Systems. Marks for each subject of all the students are provided to you. You have to tell the position of each student in the class. Print the names of each student according to their position in class. Tie is broken on the basis of their roll numbers. Between two students having same marks, the one with less roll number will have higher rank. The input is provided in order of roll number.

##### Input Format:

First line will have a single integer ‘n’, denoting the number of students in the class.

Next ‘n’ lines each will have one string denoting the name of student and three space separated integers m1, m2, m3 denoting the marks in three subjects.

##### Output Format:

Print ‘n’ lines having two values: First, the position of student in the class and second his name.

##### Constraints:

1 <= n <= 10^5

0 <= m1, m2, m3 <= 100

##### Sample Input:

3

Mohit 94 85 97

Shubham 93 91 94

Rishabh 95 81 99

##### Sample Output:

1 Shubham

2 Mohit

3 Rishabh