Codeforces Round #760 (Div. 3)

A. Polycarp and Sums of Subsequences

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp had an array a of 3 **positive** integers. He wrote out the sums of all non-empty subsequences of this array, sorted them in non-decreasing order, and got an array b of 7 integers.

For example, if $a = \{1, 4, 3\}$, then Polycarp wrote out 1, 4, 3, $1 + 4 = 5$, $1 + 3 = 4$, $4 + 3 = 7$, $1 + 4 + 3 = 8$. After sorting, he got an array $b = \{1, 3, 4, 4, 5, 7, 8\}$.

Unfortunately, Polycarp lost the array a. He only has the array b left. Help him to restore the array a.

Input

The first line contains one integer t ($1 \le t \le 5000$) — the number of test cases.

Each test case consists of one line which contains 7 integers $b_1, b_2, \dots, b_7$ ($1 \le b_i \le 10^9$; $b_i \le b_{i+1}$).

Additional constraint on the input: there exists at least one array a which yields this array b as described in the statement.

Output

For each test case, print 3 integers — a_1, a_2 and a_3. If there can be several answers, print any of them.

Example

input

```
5
1 3 4 4 5 7 8
1 2 3 4 5 6 7
300000000 300000000 300000000 600000000 600000000 600000000 900000000
1 1 2 999999998 999999999 999999999 1000000000
1 2 2 3 3 4 5
```

output

```
1 4 3
4 1 2
300000000 300000000 300000000
999999998 1 1
1 2 2
```

Note

The subsequence of the array a is a sequence that can be obtained from a by removing zero or more of its elements.

Two subsequences are considered different if index sets of elements included in them are different. That is, the values of the elements don't matter in the comparison of subsequences. In particular, any array of length 3 has exactly 7 different non-empty subsequences.

B. Missing Bigram

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp has come up with a new game to play with you. He calls it "A missing bigram".

A *bigram* of a word is a sequence of two adjacent letters in it.

For example, word "abbaaba" contains bigrams "ab", "bb", "ba", "aa", "ab" and "ba".

The game goes as follows. First, Polycarp comes up with a word, consisting only of lowercase letters 'a' and 'b'. Then, he writes down all its bigrams on a whiteboard **in the same order as they appear in the word**. After that, he wipes one of them off the whiteboard.

Finally, Polycarp invites you to guess what the word that he has come up with was.

Your goal is to find any word such that it's possible to write down all its bigrams and remove one of them, so that the resulting sequence of bigrams is the same as the one Polycarp ended up with.

The tests are generated in such a way that the answer exists. If there are multiple answers, you can print any of them.

Input

The first line contains a single integer t ($1 \le t \le 2000$) — the number of testcases.

The first line of each testcase contains a single integer n ($3 \le n \le 100$) — the length of the word Polycarp has come up with.

The second line of each testcase contains $n-2$ bigrams of that word, separated by a single space. Each bigram consists of two letters, each of them is either 'a' or 'b'.

Additional constraint on the input: there exists at least one string such that it is possible to write down all its bigrams, except one, so that the resulting sequence is the same as the sequence in the input. In other words, the answer exists.

Output

For each testcase print a word, consisting of n letters, each of them should be either 'a' or 'b'. It should be possible to write down all its bigrams and remove one of them, so that the resulting sequence of bigrams is the same as the one Polycarp ended up with.

The tests are generated in such a way that the answer exists. If there are multiple answers, you can print any of them.

Example

input

```
4
7
ab bb ba aa ba
7
ab ba aa ab ba
3
aa
5
bb ab bb
```

output

```
abbaaba
abaabaa
baa
bbabb
```

Note

The first two testcases from the example are produced from the word "abbaaba". As listed in the statement, it contains bigrams "ab", "bb", "ba", "aa", "ab" and "ba".

In the first testcase, the 5-th bigram is removed.

In the second testcase, the 2-nd bigram is removed. However, that sequence could also have been produced from the word "abaabaa". It contains bigrams "ab", "ba", "aa", "ab", "ba" and "aa". The missing bigram is the 6-th one.

In the third testcase, all of "baa", "aab" and "aaa" are valid answers.

C. Paint the Array

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an array a consisting of n positive integers. You have to choose a positive integer d and paint all elements into two colors. All elements which are divisible by d will be painted red, and all other elements will be painted blue.

The coloring is called beautiful if there are no pairs of adjacent elements with the same color in the array. Your task is to find any value of d which yields a beautiful coloring, or report that it is impossible.

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of testcases.

The first line of each testcase contains one integer n ($2 \le n \le 100$) — the number of elements of the array.

The second line of each testcase contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^{18}$).

Output

For each testcase print a single integer. If there is no such value of d that yields a beautiful coloring, print 0. Otherwise, print any suitable value of d ($1 \le d \le 10^{18}$).

Example

input

```
5
5
1 2 3 4 5
3
10 5 15
```

```
3
100 10 200
10
9 8 2 6 6 2 8 6 5 4
2
1 3
```

output

```
2
0
100
0
3
```

D. Array and Operations

time limit per test: 2 seconds

memory limit per test: 512 megabytes

input: standard input

output: standard output

You are given an array a of n integers, and another integer k such that $2k \le n$.

You have to perform **exactly** k operations with this array. In one operation, you have to choose two elements of the array (let them be a_i and a_j; they can be equal or different, but **their positions in the array must not be the same**), remove them from the array, and add $\lfloor \frac{a_i}{a_j} \rfloor$ to your *score*, where $\lfloor \frac{x}{y} \rfloor$ is the maximum integer not exceeding $\frac{x}{y}$.

Initially, your *score* is 0. After you perform exactly k operations, you add all the remaining elements of the array to the *score*.

Calculate the minimum possible *score* you can get.

Input

The first line of the input contains one integer t ($1 \le t \le 500$) — the number of test cases.

Each test case consists of two lines. The first line contains two integers n and k ($1 \le n \le 100$; $0 \le k \le \lfloor \frac{n}{2} \rfloor$).

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$).

Output

Print one integer — the minimum possible *score* you can get.

Example

input

```
5
7 3
1 1 1 2 1 3 1
5 1
5 5 5 5 5
4 2
1 3 3 7
2 0
4 2
9 2
1 10 10 1 10 2 7 10 3
```

output

```
2
16
0
6
16
```

Note

Let's consider the example test.

In the first test case, one way to obtain a score of 2 is the following one:

1. choose $a_7 = 1$ and $a_4 = 2$ for the operation; the score becomes $0 + \lfloor \frac{1}{2} \rfloor = 0$, the array becomes $[1, 1, 1, 1, 3]$;
2. choose $a_1 = 1$ and $a_5 = 3$ for the operation; the score becomes $0 + \lfloor \frac{1}{3} \rfloor = 0$, the array becomes $[1, 1, 1]$;
3. choose $a_1 = 1$ and $a_2 = 1$ for the operation; the score becomes $0 + \lfloor \frac{1}{1} \rfloor = 1$, the array becomes $[1]$;
4. add the remaining element 1 to the score, so the resulting score is 2.

In the second test case, no matter which operations you choose, the resulting score is 16.

In the third test case, one way to obtain a score of 0 is the following one:

1. choose $a_1 = 1$ and $a_2 = 3$ for the operation; the score becomes $0 + \lfloor \frac{1}{3} \rfloor = 0$, the array becomes $[3, 7]$;
2. choose $a_1 = 3$ and $a_2 = 7$ for the operation; the score becomes $0 + \lfloor \frac{3}{7} \rfloor = 0$, the array becomes empty;
3. the array is empty, so the score doesn't change anymore.

In the fourth test case, no operations can be performed, so the score is the sum of the elements of the array: $4 + 2 = 6$.

E. Singers' Tour

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

n towns are arranged in a circle sequentially. The towns are numbered from 1 to n in clockwise order. In the i-th town, there lives a singer with a repertoire of a_i minutes for each $i \in [1, n]$.

Each singer visited all n towns in clockwise order, starting with the town he lives in, and gave exactly one concert in each town. In addition, in each town, the i-th singer got inspired and came up with a song that lasts a_i minutes. The song was added to his repertoire so that he could perform it in the rest of the cities.

Hence, for the i-th singer, the concert in the i-th town will last a_i minutes, in the $(i+1)$-th town the concert will last $2 \cdot a_i$ minutes, ..., in the $((i+k) \bmod n + 1)$-th town the duration of the concert will be $(k+2) \cdot a_i$, ..., in the town $((i+n-2) \bmod n + 1)$ — $n \cdot a_i$ minutes.

You are given an array of b integer numbers, where b_i is the total duration of concerts in the i-th town. Reconstruct any correct sequence of **positive** integers a or say that it is impossible.

Input

The first line contains one integer t $(1 \le t \le 10^3)$ — the number of test cases. Then the test cases follow.

Each test case consists of two lines. The first line contains a single integer n $(1 \le n \le 4 \cdot 10^4)$ — the number of cities. The second line contains n integers $b_1, b_2, \dots, b_n$ $(1 \le b_i \le 10^9)$ — the total duration of concerts in i-th city.

The sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print the answer as follows:

If there is no suitable sequence a, print NO. Otherwise, on the first line print YES, on the next line print the sequence $a_1, a_2, \dots, a_n$ of n integers, where a_i $(1 \le a_i \le 10^9)$ is the initial duration of repertoire of the i-th singer. If there are multiple answers, print any of them.

Example

input

```
4
3
12 16 14
1
1
3
1 2 3
6
81 75 75 93 93 87
```

output

```
YES
3 1 3
YES
1
NO
YES
5 5 4 1 4 5
```

Note

Let's consider the 1-st test case of the example:

1. the 1-st singer in the 1-st city will give a concert for 3 minutes, in the 2-nd — for 6 minutes, in the 3-rd — for 9 minutes;
2. the 2-nd singer in the 1-st city will give a concert for 3 minutes, in the 2-nd — for 1 minute, in the 3-rd - for 2 minutes;
3. the 3-rd singer in the 1-st city will give a concert for 6 minutes, in the 2-nd — for 9 minutes, in the 3-rd — for 3 minutes.

F. Reverse

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given two positive integers x and y. You can perform the following operation with x: write it in its binary form without leading zeros, add 0 or 1 to the right of it, reverse the binary form and turn it into a decimal number which is assigned as the new value of x.

For example:

- 34 can be turned into 81 via one operation: the binary form of 34 is 100010, if you add 1, reverse it and remove leading zeros, you will get 1010001, which is the binary form of 81.
- 34 can be turned into 17 via one operation: the binary form of 34 is 100010, if you add 0, reverse it and remove leading zeros, you will get 10001, which is the binary form of 17.
- 81 can be turned into 69 via one operation: the binary form of 81 is 1010001, if you add 0, reverse it and remove leading zeros, you will get 1000101, which is the binary form of 69.
- 34 can be turned into 69 via two operations: first you turn 34 into 81 and then 81 into 69.

Your task is to find out whether x can be turned into y after a certain number of operations (possibly zero).

Input

The only line of the input contains two integers x and y ($1 \le x, y \le 10^{18}$).

Output

Print YES if you can make x equal to y and NO if you can't.

Examples

input

```
3 3
```

output

```
YES
```

input

```
7 4
```

output

```
NO
```

input

```
2 8
```

output

```
NO
```

input

```
34 69
```

output

```
YES
```

input

```
8935891487501725 71487131900013807
```

output

```
YES
```

Note

In the first example, you don't even need to do anything.

The fourth example is described in the statement.

G. Trader Problem

time limit per test: 4.5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Monocarp plays a computer game (yet again!). This game has a unique trading mechanics.

To trade with a character, Monocarp has to choose one of the items he possesses and trade it for some item the other character possesses. Each item has an integer price. If Monocarp's chosen item has price x, then he can trade it for any item **(exactly one item)** with price not greater than $x + k$.

Monocarp initially has n items, the price of the i-th item he has is a_i. The character Monocarp is trading with has m items, the price of the i-th item they have is b_i. Monocarp can trade with this character as many times as he wants (possibly even zero times), each time exchanging one of his items with one of the other character's items according to the aforementioned constraints. Note that if Monocarp gets some item during an exchange, he can trade it for another item (since now the item belongs to him), and vice versa: if Monocarp trades one of his items for another item, he can get his item back by trading something for it.

You have to answer q queries. Each query consists of one integer, which is the value of k, and asks you to calculate the maximum possible total cost of items Monocarp can have after some sequence of trades, assuming that he can trade an item of cost x for an item of cost not greater than $x + k$ during each trade. Note that the queries are independent: the trades do not actually occur, Monocarp only wants to calculate the maximum total cost he can get.

Input

The first line contains three integers n, m and q ($1 \le n, m, q \le 2 \cdot 10^5$).

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$) — the prices of the items Monocarp has.

The third line contains m integers $b_1, b_2, \dots, b_m$ ($1 \le b_i \le 10^9$) — the prices of the items the other character has.

The fourth line contains q integers, where the i-th integer is the value of k for the i-th query ($0 \le k \le 10^9$).

Output

For each query, print one integer — the maximum possible total cost of items Monocarp can have after some sequence of trades, given the value of k from the query.

Example

input

```
3 4 5
10 30 15
12 31 14 18
0 1 2 3 4
```

output

```
55
56
60
64
64
```