Problema C - Remoções Aproximadas

Rita has a multiset of integers $\boldsymbol{A}$, and a set of integers $\boldsymbol{B}$. She wants to remove some elements from $\boldsymbol{A}$ in such a way that, in the multiset that remains, it never happens that the sum of two elements belongs to $\boldsymbol{B}$.

More precisely, after the removals, for any two remaining elements x and y (they may be equal), it must hold that $x + y$ **does not** belong to $\boldsymbol{B}$. In particular, it is allowed to form a pair by using the same remaining element twice, even if that value appears only once in $\boldsymbol{A}$.

Finding the minimum number of removals is difficult. Therefore, in this problem it is enough to present an *approximate* solution:

- let $\boldsymbol{OPT}$ be the minimum number of elements that must be removed to make $\boldsymbol{A}$ valid;
- your submission is accepted if it prints a valid removal with at most $2 \cdot \boldsymbol{OPT}$ elements.

If a value appears several times in $\boldsymbol{A}$, each occurrence may be removed separately. However, if you print a value t times in the output of a given test case, that value must appear at least t times in the input of that test case.

There are $\boldsymbol{T}$ independent test cases and for each one you must print a solution.

Example

Consider an example where $\boldsymbol{A} = \{1, 2, 3, 7, 10\}$ and $\boldsymbol{B} = \{4, 13\}$.

If we remove $\{2, 3\}$ from $\boldsymbol{A}$, we are left with the values $\{1, 7, 10\}$.

The possible sums between remaining values are $1 + 1 = 2$, $1 + 7 = 8$, $1 + 10 = 11$, $7 + 7 = 14$, $7 + 10 = 17$ and $10 + 10 = 20$. None of them belongs to $\boldsymbol{B} = \{4, 13\}$, so the removal is valid.

Constraints

The following limits are guaranteed in all test cases that will be given to the program:

$1 \leq \boldsymbol{T} \leq 1000$	number of test cases
$1 \leq \boldsymbol{N_i} \leq 10^5$	number of elements of $\boldsymbol{A}$ in case i
$1 \leq \boldsymbol{M_i} \leq 100$	number of elements of $\boldsymbol{B}$ in case i
$\sum_{i=1}^{\boldsymbol{T}} \boldsymbol{N_i} \leq 10^5$	sum of all values of $\boldsymbol{N_i}$
$-10^9 \leq a_{i,j} \leq 10^9$	value of the j-th element of $\boldsymbol{A}$ in case i
$-2 \cdot 10^9 \leq b_{i,j} \leq 2 \cdot 10^9$	value of the j-th element of $\boldsymbol{B}$ in case i
$b_{i,j} \neq b_{i,k}$ for $j \neq k$	all values of $\boldsymbol{B}$ are distinct within each case

The test cases are organized into the following groups:

Subtask	Points	Additional Constraints
1	20	$\sum_{i=1}^{\boldsymbol{T}} \boldsymbol{N_i} \leq 20$
2	20	$\boldsymbol{M_i} = 1$ in every case
3	25	In each case, all values of $\boldsymbol{A}$ are distinct
4	35	No additional constraints

Input Format

The first line contains an integer $\boldsymbol{T}$: the number of test cases.

Then $\boldsymbol{T}$ test cases follow. Each test case has the following format:

- one line with two integers $\boldsymbol{N}$ and $\boldsymbol{M}$, separated by a space: the number of elements of $\boldsymbol{A}$ and the number of elements of $\boldsymbol{B}$;
- one line with $\boldsymbol{N}$ integers $a_1, a_2, \ldots, a_{\boldsymbol{N}}$, separated by spaces: the elements of $\boldsymbol{A}$;
- one line with $\boldsymbol{M}$ integers $b_1, b_2, \ldots, b_{\boldsymbol{M}}$, separated by spaces: the elements of $\boldsymbol{B}$.

Output Format

For each test case, in the same order as the input, print two lines:

- on the first line, an integer $\boldsymbol{k}$: the number of elements that you decide to remove in that case;
- on the second line, $\boldsymbol{k}$ integers, each equal to a value present in $\boldsymbol{A}$ of that case, describing the removed multiset.

If a value x appears t times in the output of a case, then x must appear at least t times in the input of that case.

Your answer for a given case is accepted if:

- after removing exactly those occurrences, no pair x, y remains (possibly with $x = y$) such that $x + y$ belongs to $\boldsymbol{B}$; the same remaining element may be used in both positions of the pair;

- $k \leq 2 \cdot OPT$, where OPT is the minimum possible number of removals in that case.

Note on the following example: since this is an approximation problem, the shown output does not have to be optimal. In this case, $OPT = 1$, but the answer with 2 removals is still accepted.

Example 1 Input

```
2
5 2
1 2 3 7 10
4 13
4 1
5 5 5 5
10
```

Example 1 Output

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