Google Code Jam '14 Qualification Round Problem C - Deceitful War

Time limit: 120.0s **Memory limit:** 1G

Naomi and Ken sometimes play games together. Before they play, each of them gets N identical-looking blocks of wood with masses between 0.0kg and 1.0kg (exclusive). All of the blocks have different weights. There are lots of games they could play with those blocks, but they usually play something they call War. Here is how War works:

1. Each player weighs each of his or her own blocks, so each player knows the weights of all of his or her own blocks, but not the weights of the other player's blocks.
2. They repeat the following process N times:
    1. Naomi chooses one of her own blocks, with mass $Chosen_{Naomi}$.
    2. Naomi tells Ken the mass of the block she chose.
    3. Ken chooses one of his own blocks, with mass $Chosen_{Ken}$.
    4. They each put their block on one side of a balance scale, and the person whose block is heavier gets one point.
    5. Both blocks are destroyed in a fire.

Naomi has realized three things about War. First, she has realized that she loses a lot. Second, she has realized that there is a unique strategy that Ken can follow to maximize his points without assuming anything about Naomi's strategy, and that Ken always uses it. Third, she has realized that she hates to lose. Naomi has decided that instead of playing War, she will play a game she calls Deceitful War. The great thing about Deceitful War is that Ken will think they're playing War!

Here is how Deceitful War works, with differences between Deceitful War and War in bold:

1. Each player weighs each of his or her own blocks. **Naomi also weighs Ken's blocks while he isn't looking, so Naomi knows the weights of all blocks** and Ken only knows the weights of his own blocks.
2. They repeat the following process N times:
    1. Naomi chooses one of her own blocks, with mass $Chosen_{Naomi}$.
    2. Naomi tells Ken **a number, $Told_{Naomi}$, between 0.0kg and 1.0kg exclusive.** Ken, who thinks they're playing War, thinks the number Naomi just told him is $Chosen_{Naomi}$.
    3. Ken chooses one of his own blocks, with mass $Chosen_{Ken}$.
    4. They each put their block on one side of a balance scale, and the person whose block is heavier gets one point.
    5. Both blocks are destroyed in a fire.

Naomi doesn't want Ken to know that she isn't playing War; so when she is choosing which block to play, and what mass to tell Ken, she must make sure that the balance scale won't reveal that $Chosen_{Naomi} \neq Told_{Naomi}$. In other words, she must make decisions so that:

- $Chosen_{Naomi} > Chosen_{Ken}$ if, and only if, $Told_{Naomi} > Chosen_{Ken}$, and
- $Told_{Naomi}$ is not equal to the mass of any of Ken's blocks, because he knows that isn't possible

It might seem like Naomi won't win any extra points by being deceitful, because Ken might discover that she wasn't playing War; but Naomi knows Ken thinks both players are playing War, and she knows what he knows, and she knows Ken will always follow his unique optimal strategy for War, so she can always predict what he will play.

You'll be given the masses of the blocks Naomi and Ken started with. Naomi will play Deceitful War optimally to gain the maximum number of points. Ken will play War optimally to gain the maximum number of points *assuming that both players are playing War*. What will Naomi's score be? What would it have been if she had played War optimally instead?

Examples

If each player has a single block left, where Naomi has 0.5kg and Ken has 0.6kg, then Ken is guaranteed to score the point. Naomi can't say her number is ≥ 0.6kg, or Ken will know she isn't playing War when the balance scale shows his block was heavier.

If each player has two blocks left, where Naomi has $[0.7\text{kg}, 0.2\text{kg}]$ and Ken has $[0.8\text{kg}, 0.3\text{kg}]$, then Naomi could choose her 0.2kg block, and deceive Ken by telling him that she chose a block that was 0.6kg. Ken assumes Naomi is telling the truth (as in how the War game works) and will play his 0.8kg block to score a point. Ken was just deceived, but he will never realize it because the balance scale shows that his 0.8kg block is, like he expected, heavier than the block Naomi played. Now Naomi can play her 0.7kg block, tell Ken it is 0.7kg, and score a point. If Naomi had played War instead of Deceitful War, then Ken would have scored two points and Naomi would have scored zero.

Input Specification

The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with a line containing a single integer N, the number of blocks each player has. Next follows a line containing N space-separated real numbers: the masses of Naomi's blocks, in kg. Finally there will be a line containing N space-separated real numbers: the masses of Ken's blocks, in kg.

Each of the masses given to Ken and Naomi will be represented as a 0, followed by a decimal point, followed by 1-5 digits. Even though all the numbers in the input have 1-5 digits after the decimal point, Ken and Naomi don't know that; so Naomi can still tell Ken that she played a block with mass 0.5000001kg, and Ken has no reason not to believe her.

Output Specification

For each test case, output one line containing `Case #x: y z`, where x is the test case number (starting from 1), y is the number of points Naomi will score if she plays Deceitful War optimally, and z is the number of points Naomi will score if she plays War optimally.

Limits

Memory limit: 1 GB.

$1 \leq T \leq 50$.

All the masses given to Ken and Naomi are distinct, and between 0.0 and 1.0 exclusive.

Small dataset

Time limit: 60 seconds.

$1 \leq N \leq 10$.

Large dataset

Time limit: 120 seconds.

$1 \leq N \leq 1\,000$.

Sample Input

```
4
1
0.5
0.6
2
0.7 0.2
0.8 0.3
3
0.5 0.1 0.9
0.6 0.4 0.3
9
0.186 0.389 0.907 0.832 0.959 0.557 0.300 0.992 0.899
0.916 0.728 0.271 0.520 0.700 0.521 0.215 0.341 0.458
```

Sample Output

```
Case #1: 0 0
Case #2: 1 0
Case #3: 2 1
Case #4: 8 4
```

Note

This problem has different time limits for different batches. If you exceed the Time Limit for any batch, the judge will incorrectly display `>120.000s` regardless of the actual time taken. Refer to the **Limits** section for batch-specific time limits.