Code Jam 2020 - Round 3

Naming Compromise

Problem

Cameron and Jamie are about to welcome a second baby into their lives. They are already good at working together as parents, but right now they are disagreeing about one crucial thing! Cameron wants to name the baby one name (the string **C**), whereas Jamie wants to name the baby something else (the string **J**).

You want to help them find a *compromise name* that is as close as possible to what each of them wants. You think you can do this using the notion of *edit distance*. The edit distance between two strings S_1 and S_2 is the minimum number of operations required to transform S_1 into S_2, where the allowed operations are as follows:

- Insert one character anywhere in the string.
- Delete one character from anywhere in the string.
- Change one character in the string to any other character.

For example, the edit distance between `CAMERON` and `JAMIE` is 5. One way to accomplish the transformation in 5 steps is the following: `CAMERON` to `JAMERON` (change) to `JAMIERON` (insert) to `JAMIEON` (delete) to `JAMIEN` (delete) to `JAMIE` (delete). Any transformation from `CAMERON` into `JAMIE` requires at least this many operations.

To make the compromise name N as close as possible to the original desires of the parents, you want N to be a non-empty string such that the sum of the edit distances between **C** and N and between **J** and N is as small as possible. Out of all those choices for N, to make sure the compromise is fair, you must choose an N such that the difference between those two edit distances is also as small as possible. Please find a compromise name for Cameron and Jamie.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each case consists of a single line with two strings **C** and **J**: the names that Cameron and Jamie have proposed for the baby, respectively. Each of these names is made up of uppercase English alphabet letters.

Output

For each test case, output one line containing `Case #x: y`, where `x` is the test case number (starting from 1) and `y` is a name that meets the requirements mentioned in the statement. Note that `y` must contain only uppercase English letters.

Limits

Time limit: 20 seconds per test set.
Memory limit: 1GB.
$1 \leq \mathbf{T} \leq 100$.
$\mathbf{C} \neq \mathbf{J}$.

Test Set 1 (Visible Verdict)

1 ≤ length of **C** ≤ 6.
1 ≤ length of **J** ≤ 6.
The i-th letter of **C** is an uppercase `X`, `Y`, or `Z`, for all i.
The i-th letter of **J** is an uppercase `X`, `Y`, or `Z`, for all i.

Test Set 2 (Hidden Verdict)

1 ≤ length of **C** ≤ 60.
1 ≤ length of **J** ≤ 60.
The i-th letter of **C** is an uppercase English alphabet letter, for all i.
The i-th letter of **J** is an uppercase English alphabet letter, for all i.

Sample

Sample Input

```
4
XYZZY ZZYZX
Y Z
YYXXYZ ZYYXXY
XZXZXZ YZ
```

Sample Output

```
Case #1: ZZY
Case #2: Z
Case #3: ZYYXXYZ
Case #4: ZYZX
```

The above cases meet the limits for Test Set 1. Another sample case that does not meet those limits appears at the end of this section.

In Sample Case #1, the edit distance from `XYZZY` to `ZZY` is 2 (delete the first two letters), and the edit distance from `ZZYZX` to `ZZY` is 2 (delete the last two letters). `XZZX` and `ZYYZY` would also work. No possible name has a sum of edit distances that is less than 4.

`ZY`, for example, has the same edit distance to **C** as to **J** (3, in each case). However the sum of those distances would be 6, which is not minimal, so it would not be an acceptable answer.

`XZZY` is also unacceptable. Its edit distances to **C** and **J**, respectively, are 1 and 3. The sum of those edit distances is minimal, but the difference between the two (|1-3| = 2) is not minimal, since we have shown that it is possible to achieve a difference of 0.

In Sample Case #2, `Y` and `Z` are the only acceptable answers.

In Sample Case #3, notice that input length restrictions do not apply to the output, so the shown answer is acceptable in either test set. Another possible answer is `YYXXY`.

In Sample Case #4, the edit distance between `XZXZXZ` and `ZYZX` is 3, and the edit distance between `YZ` and `ZYZX` is 2. The sum of those edit distances is 5, and their difference is 1; these values are optimal for this case.

The following additional case could not appear in Test Set 1, but could appear in Test Set 2.

```
1
GCJ ABC
```

`Case #1: GC` is one of the possible correct outputs.