We are in desperate need of expert hands — hands that can open combination locks in a flash. We already know the initial and target state of the lock, but we need someone who can turn the wheels using the smallest number of steps needed to open the locks fast.

A multi-wheel combination lock has multiple wheels that one can turn either clockwise or counter clockwise to pick a digit from 0 to 9. The digits wrap around, so you get digits 0 right after you cross 9 in a clockwise fashion. Similarly, you get 9 after you cross 0 in a counter clockwise fashion.

Given the number of wheels on a combination lock, its initial state and the target state, you need to find out the minimum number of steps needed to go from the initial state to the target state. For every change in a digit on a wheel, we will count one step. For example, for a 4 wheel combination lock if we are given the initial configuration "1234" and we want to get to "2546" we need at least 7 steps (1 step to go from 1 to 2, 3 steps to go from 2 to 5, 1 step to go from 3 to 4 and 2 steps to go from 4 to 6).

Input

The first line of input will contain an integer T ($1 \leq T \leq 200$), the number of test cases. Each of the following T lines will give you n, *initial*, and *target*. Here n ($1 \leq n \leq 100$) is an integer specifying the number of wheels in the combination lock. The numeric strings *initial* and *target* denotes the initial and target states of the lock. The length of both strings will be exactly n and they will only contain decimal digits.

Output

For each line of input, print the case number as '`Case` X`:` Y' where X is the case number (starting from 1) followed by Y, the minimum number of steps needed to go from the initial configuration to the target configuration. Check sample input and output for details.

Sample Input

```
2
4 1234 2546
4 1234 1235
```

Sample Output

```
Case 1: 7
Case 2: 1
```