Broken Swords

Ken is a fencer with a big problem: he swings too hard!

Whenever Ken gets a new sword, he's sure to break it sooner rather than later. It's occurred to him that this habit is costing him quite a bit of money, since he normally buys a new sword whenever his current sword breaks. However, he's heard from a fellow club member that he could instead make new swords out of his old swords!


Source: Dan Folescu

Each bamboo sword Ken uses has four bamboo 'slats' which help to cushion each blow for the receiving partner. Whenever Ken breaks his sword, he breaks either one, two, three, or all four slats.

When Ken first tried to put a sword together with the remains of two broken swords, he realized that not all slats are equivalent. When the slats are bundled together, they conform to the shape of the slat which sits opposite!

Ken doesn't want an uncomfortable grip, so he will put only an old slat in a new position if it is of the same or opposite type. There are four possible slats "top" (T), "bottom" (B), "left" (L), and "right" (R). When looking at a sword head on, these slats form the following configuration:
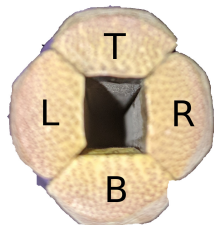
As an example, Ken will only replace a top slat with a top or bottom slat.

After collecting broken swords for quite a while, Ken has decided to build as many swords as possible. Write a program to compute how many swords he will be able to build and how many slats he'll have left over!

Input

The input consists of a single test case. On the first line, you are given N, the number of swords Ken has broken. On each of the next N lines, you are given four characters. The characters correspond to the following slat ordering: TBLR. If the character in a position is 0, then the slat is not broken, and if it is a 1, Ken is out of luck!

Output

Output three numbers: the total number of swords Ken can make, the sum of the numbers of the remaining T and B slats, and the sum of the numbers of the remaining L and R slats, respectively.

Sample Input 1

```
4
0100
0010
0110
1010
```

Sample Output 1

```
2 1 1
```

Hide ›

Please log in to submit a solution to this problem

Log in