Your brilliant but absent-minded uncle believes he has solved a difficult crossword puzzle but has misplaced the solution. He needs your help to reconstruct the solution from a list that contains all the words in the solution, plus one extra word that is not part of the solution. Your program must solve the puzzle and print the extra word.

The crossword puzzle is represented by a grid with ten squares on each side. Figure 1 shows the top left corner of a puzzle. The puzzle has a certain number of "slots" where a word can be placed. Each slot is represented by the row and column number of the square where the slot begins, and the direction in which the slot extends from its initial square ("across" or "down"). The length of each slot is not specified. The puzzle has a list of candidate words, all but one of which is used in solving the puzzle.

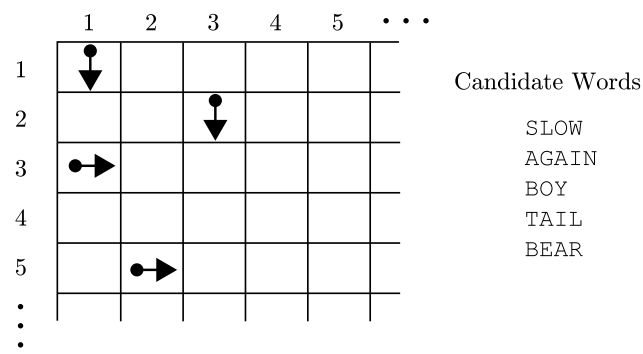

Figure 1: Corner of Example Puzzle

Figure 2 shows a solution to the example puzzle in Figure 1. In a valid solution, each slot is filled with a candidate word. The word at a slot is defined as the maximal horizontal/vertical sequence from the origin of the slot, along its direction. Further more, the previous grid of the origin along the direction must not be a letter (which would be confusing). The set of words at all slots must be the set of input words excluding an extra word.

Any candidate word can be used in any slot as long as the word fits in the puzzle and does not conflict with any other word. In the example, all the candidate words are used except the word "BOY".

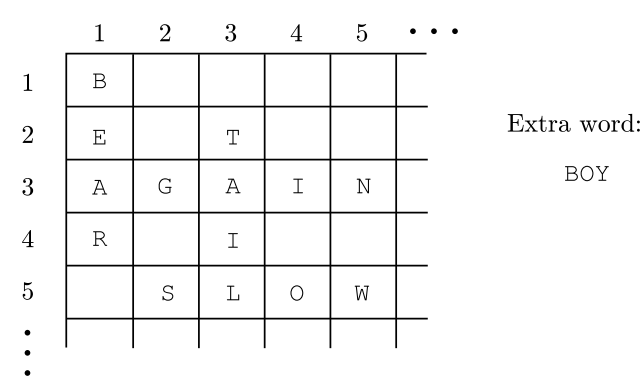

Figure 2: Example Solution

Input

The input data consist of one or more test cases each describing a puzzle trial. The first input line in each test case contains a positive integer N that represents the number of slots in the puzzle. This line is followed by N lines, each containing the row number and column number of a square where a slot begins, followed by the letter '`A`' (if the slot is "Across") or '`D`' (if the slot is "Down"). The next $N+1$ input lines contain candidate words that can be used in the puzzle solution. Words are mutually different. All the words have at least two characters.

The final test case is followed by a line containing the number zero.

Output

For each trial, print the trial number followed by the word that is not used in the puzzle solution, using the format in the example output. Observe the following rules:

1. Print a blank line after each trial.
2. If your uncle has made a mistake and the puzzle has no solution using the given words, print the word '`Impossible`'. For example, if Trial 2 has no solution, you should print '`Trial 2: Impossible`'.
3. If the puzzle can be solved in more than one way, print each word that can be omitted from a valid solution. The words can be printed in any order but each word must be printed only once. For example, if Trial 3 has a solution that omits the word '`DOG`' and two solutions that omit the word '`CAT`', you should print '`Trial 3:  DOG CAT`' or '`Trial 3:  CAT DOG`'.

Sample Input

```
4
1   1   D
2   3   D
3   1   A
5   2   A
SLOW
AGAIN
BOY
TAIL
BEAR
0
```

Sample Output

```
Trial 1: BOY
```