

**2011-0920 UM Online Practice Programming Contest
September 20, 2011**

**Sponsored by:
University of Michigan**

Rules:

1. There are a few questions to be solved any time during the tryouts contest.
2. All questions require you to read the test data from standard input and write results to standard output. You cannot use files for input or output. Additional input and output specifications can be found in the General Information Sheet.
3. The allowed programming languages are C, C++ and Java.
4. All programs will be re-compiled prior to testing with the judges' data.
5. Non-standard libraries cannot be used in your solutions. The Standard Template Library (STL) and C++ string libraries are allowed. The standard Java API is available, except for those packages that are deemed dangerous by contestant officials (e.g., that might generate a security violation).
6. Programming style is not considered in this contest. You are free to code in whatever style you prefer. Documentation is not required.
7. All communication with the judges will be handled by the PC2 environment.
8. Judges' decisions are to be considered final. No cheating will be tolerated.

Problem A: Pass Feil!

Todd Feil is one of the ICPC Judges. Really. Watch out, as he will try to fail you now, because one day he discovered that his class has an unusual amount of students named Bartholomew. To find out just exactly how many Barts are in his class, he wrote down each student's name on a sheet of paper and got the following list:

John, Bart, Michael, Bart, John, Bart, Peter, Eugene, Dave, David, Bart

Then, he counted up the number of Barts he wrote down, and he was right! There were 4 Bart students, more Barts than students with any other name in the class. "Hmm", said Professor Feil to himself hardly trying to suppress evil cackling, "It will be most interesting to find out the most popular student name in the entire school and also, how many students have that name". The longer he thought about this the more he wanted to know the answer. He knew it will be a lot of work, since school has more than 2,000 people! The more he thought about it the more tedious it became. Professor Feil realized that he fell short of his own expectations. Recalling that his class has a very bright student – you! – he now seeks your help. Professor wants you to write a program that takes a list of names, and figures out how many students on campus have the most popular name and what that name is. And if you don't come up with the answer, he will surely fail you and you will have to put up with Prof. Feil's smugly-triumphal smirk for the rest of the semester. Clearly, you better get right on this.

Input

Input will consist of multiple cases. Each line will contain an integer n between 1 and 2,500 inclusively, where n is the number of student names to follow, followed by n student names separated by a space. Each name will be no longer than 20 characters and consist of characters 'a' through 'z' with the first letter capitalized. Last line will start with 0, and should not be processed. You are guaranteed that there will be a unique most popular name for every case.

Output

For each line of input professor expects you to print out the most popular name in the list, followed by the number of how many students have that name, with a space in between. The name should be capitalized as it was in the original list.

Sample Input

```
11 John Bart Michael Bart John Bart Peter Eugene Dave David Bart
10 Mike Mark Dennis Raiden Smoke Belokk Goro Kevin Mike Andy
0
```

Sample Output

```
Bart 4
Mike 2
```


- You can rotate the board only once.
- Assume that gravity only takes effect after the board has been rotated completely.
- Only check for winners after gravity has finished taking effect.

Input

The first line of the input gives the number of test cases, T , $1 \leq T \leq 100$. T test cases follow, each beginning with a line containing the integers N , $3 \leq N \leq 50$ and K , $3 \leq K \leq N$. The next N lines will each be exactly N characters long, showing the initial position of the board, using the same format as the diagrams above.

The initial position in each test case will be a legal position that can occur during a game of Join- K . In particular, neither player will have already formed K pieces in a row.

Output

For each test case, output one line containing "Case #x: y", where x is the case number starting from 1, and y is one of "Red", "Blue", "Neither", or "Both". Here, y indicates which player or players will have K pieces in a row after you rotate the board.

Sample Input

```
4
7 3
.....
.....
.....
...R...
...BB..
..BRB..
.RRBR..
6 4
.....
.....
.R...R
.R...BB
.R.RBR
RB.BBB
4 4
R...
BR..
BR..
BR..
3 3
B..
RB.
RB.
```

Sample Output

```
Case #1: Neither
Case #2: Both
Case #3: Red
Case #4: Blue
```

Problem C: FreeCell Stats

Mark played D ($D > 0$) games of FreeCell today. Each game of FreeCell ends in one of two ways – Mark either wins, or loses. Mark has been playing for many years, and has so far played G games in total (naturally, $G \geq D$).

At the end of the day, Mark looks at the game statistics to see how well he played. It turns out that he won exactly P_D percent of the D games today, and exactly P_G percent of G total games he had ever played. Miraculously, there is no rounding necessary -- both percentages are exact! Unfortunately, Mark does not remember the exact number of games he played today (D), or the exact number of games that he played in total (G). He does know that he could not have played more than N games today ($D \leq N$).

Are the percentages displayed possible, or is the game statistics calculator broken?

Input

The first line of the input gives the number of test cases, T , $1 \leq T \leq 2000$. T lines follow. Each line contains 3 integers – N , $1 \leq N \leq 10^{15}$, P_D , $0 \leq P_D \leq 100$ and P_G , $0 \leq P_G \leq 100$.

Output

For each test case, output one line containing "Case #x: y", where x is the case number (starting from 1) and y is either "Possible" or "Broken".

Sample Input

```
3
1 100 50
10 10 100
9 80 56
```

Sample Output

```
Case #1: Possible
Case #2: Broken
Case #3: Possible
```

Sample Explanation

In Case #3, Mark could have played 5 games today ($D = 5$) and 25 games in total ($G = 25$), and won 4 games today (80% of 5) and 14 games in total (56% of 25). Note: $O(N)$ algorithm might be slow.

Problems D and E are currently removed until October 1st due to them being used in another event.

Ask dennismv if you need the full PDF.