Problem

After many sleepless nights, you have finally finished teaching a robotic arm to make the hand gestures required for he Rock-Paper-Scissors game. Now you just need to program it to compete in the upcoming robot tournament!

In this tournament, each robot uses a program that is a series of moves, each of which must be one of the following: (for "Rock"), P (for "Paper"), or S (for "Scissors"). Paper beats Rock and loses to Scissors; Rock beats Scissors and oses to Paper; Scissors beats Paper and loses to Rock.

When two robots face off in a match, the first robot to play a winning move wins. To start, each robot plays the first ove of its program. If the two moves are different, one of the moves beats the other and thus one of the robots wins t e match. If the moves are the same, each robot plays the next move in its program, and so on.

Whenever a robot has reached the end of its program and needs its next move, it returns to the start of its program. S , for example, the fifth move of a robot with the program RSSP would be R. If a match goes on for over a googol (1 100) of moves, the judges flip a fair coin to determine the winner.

Once a match is over, the winning robot resets, so it has no memory of the that match. In its next match, it starts by laying the first move of its program, and so on.

The tournament is played in K rounds and has a single-elimination "bracket" structure. There are N = 2K robots in to al, numbered 0 through N - 1. In the first round, robot 0 plays a match against robot 1, robot 2 plays a match against obot 3, and so on, up to robots N - 2 and N - 1. The losers of those matches are eliminated from the tournament. In t e second round, the winner of the 0-1 match faces off against the winner of the 2-3 match, and so on. Once we get to the K-th round, there is only one match, and it determines the overall winner of the tournament.

All of the other contestants are so confident that they have already publicly posted their robots' programs online. Ho ever, the robots have not yet been assigned numbers, so nobody knows in advance who their opponents will be. Kno ing all of the other programs, is it possible for you to write a program that is guaranteed to win the tournament, no m tter how the robot numbers are assigned?

Input

The first line of the input gives the number of test cases, T; T test cases follow. Each test case begins with one line c ntaining an integer A: the number of adversaries (other robots) in the tournament. Then, there are A more lines; the i th of these contains a string Ci of uppercase letters that represent the program of the i-th opponent's robot.

Output

For each test case, output one line containing Case #x: y. If there is a string of between 1 and 500 characters that is g aranteed to win the tournament, as described above, then y should be the string of uppercase letters representing that rogram. Otherwise, y should be IMPOSSIBLE, in uppercase letters.

Limits

$1 \leq T \leq 100$.
Time limit: 20 seconds per test set.
Memory limit: 1GB.
Each character in Ci is uppercase R, P, or S, for all i.
$A = 2K - 1$ for some integer $K \geq 1$.

Test set 1 (Visible)

$1 \leq A \leq 7$.
Ci is between 1 and 5 characters long, for all i.

Test set 2 (Hidden)

$1 \leq A \leq 255$.
Ci is between 1 and 500 characters long, for all i.

Sample

Input

Output

3
1
RS
3
R
P
S
7
RS
RS
RS
RS
RS
RS
RS

Case #1: RSRSRSP
Case #2: IMPOSSIBLE
Case #3: P

Note: Although all the opponents in each of these sample cases have programs of the same length, this is not necessa ily the case. Opponents within a test case might have programs of different lengths.

In Sample Case #1, there is only one opponent, with the program RS. Our answer matches the opponent's moves for while, and the opponent loops through its program several times. As is starts its fourth pass through its program, we beat it with P. Other valid solutions exist, like P, RR, and R.

In Sample Case #2, there are three opponents, with the programs R, P, and S. It is up to you to figure out why this ca e is IMPOSSIBLE!

In Sample Case #3, all seven opponents use the same program. Using the program P, for example, guarantees that y u will win. Remember that each robot begins at the start of its program at the start of each match against a new oppo ent.

Solution:

```
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
```

```
using namespace std;

int main(){
 ios_base::sync_with_stdio(false);
 int num_cases;
 cin >> num_cases;
 for(int case_num = 1; case_num <= num_cases; ++case_num){
  int n;
  cin >> n;
  vector<string> vs(n);
  for(int i = 0; i < n; ++i){ cin >> vs[i]; }
  vector<bool> done(n);
  vector<char> solution;
  bool accept = true;
  for(int i = 0; accept && i < 500; ++i){
   int mask = 0;
   for(int j = 0; j < n; ++j){
    if(done[j]){ continue; }
    const auto& s = vs[j];
    const int c = s[i % s.size()];
    switch(c){
    case 'R': mask |= 1; break;
    case 'P': mask |= 2; break;
    case 'S': mask |= 4; break;
    }
   }
   if(mask == 7){
    accept = false;
    break;
   }else if(mask == 3 || mask == 1){
    solution.push_back('P');
    for(int j = 0; j < n; ++j){
     const auto& s = vs[j];
     if(s[i % s.size()] == 'R'){ done[j] = true; }
    }
   }else if(mask == 5 || mask == 4){
    solution.push_back('R');
    for(int j = 0; j < n; ++j){
     const auto& s = vs[j];
     if(s[i % s.size()] == 'S'){ done[j] = true; }
    }
   }else if(mask == 6 || mask == 2){
    solution.push_back('S');
    for(int j = 0; j < n; ++j){
     const auto& s = vs[j];
     if(s[i % s.size()] == 'P'){ done[j] = true; }
    }
   }else{
    break;
   }
  }
```

```
  cout << "Case #" << case_num << ": ";
  if(!accept){
   cout << "IMPOSSIBLE" << endl;
  }else{
   solution.push_back('\0');
   cout << solution.data() << endl;
  }
 }
 return 0;
}
```