Problem F: Baseball

Input: `baseball.in`
Output: `baseball.out`

Baseball is a game full of statistics. But what good are these statistics? Maybe they can be used to predict the outcome of a game.

A baseball game consists of 9 innings. If the game is tied after 9 innings, one additional inning is played at a time until the game is no longer tied at the end of the additional inning. Each inning is divided into two halves, in which one team "attacks" and the other team "defends". The visiting team attacks in the first half, while the home team attacks in the second half. If the home team is leading at the start of the second half of the 9th inning, the game ends because the winner has already been determined.

Each team submits a "batting order" at the beginning of the game indicating the order in which its 9 players will bat during the attack half of the inning. In each half of an inning, the players in the attacking team bat according to the batting order. The first batter in the first inning is the first one in the batting order. Each subsequent batter is the next batter in the batting order. If the previous batter is the last player in the batting order, the next batter is the first player in the batting order again. In each subsequent inning, the first player to bat is the player following the last player who has batted in the batting order.

When a player is successful at bat (a hit), the player at bat advances to the first base, and any other players already on base advance by one base (in real games a player may advance by multiple bases, but we will not consider this case in this problem). A player must advance to first base, second base, third base, and finally the home base in order to score a run. When a player reaches the home base, he is returned to the bench to wait for his next opportunity to bat. The half of an inning continues indefinitely until 3 attacking players are unsuccessful at bat (each unsuccessful batter is an "out"). When this happens, any players left on first, second, or third base return to the bench and do not score. The team that scores the most runs at the end of the game wins.

In certain situations, a player may "sacrifice" himself in order to advance the players already on base. If a sacrifice is successful, the player at bat is out but each player already on base is advanced by one base. A player advancing to the home base this way scores a run, unless the sacrificed batter is the third out of the inning. In the latter case, the half of the inning is over and no run is scored. If a sacrifice is unsuccessful, the batter is out and none of the players advances. A batter will attempt to sacrifice whenever there is a player on second base with zero out, or when there is a player on third base with at most one out.

One of the most commonly cited statistics for players is the "hit percentage" (between 0 and 1) for each player, indicating the proportion of time the player is successful at bat. Similarly, the "sacrifice percentage" of a player is the proportion of time the player is successful at a sacrifice. In this problem, you will be given the hit percentage and the sacrifice percentage of each of the 9 players in a team as well as a batting order. You will be asked to simulate a baseball game. Random numbers are required in a simulation, and you will use the following random number generator:

```
x(n+1) = (x(n) * 25173 + 13849) % 65536
```

where `x(n)` is the previous random number and `x(n+1)` is the next random number. The calculations above should be performed using 32–bit integers. You should start with the "seed" `x(0) = 1`; the first random number generated by your program is `x(1)`. Each time the simulation needs to determine if a hit or a sacrifice is successful, it should generate the next random number. A hit (or a sacrifice) is successful if

```
random_number / 65536 <= hit (or sacrifice) percentage
```

The division in this formula is floating−point division. Do not reset the random number generator (i.e. resetting the seed to 1) except at the beginning of each game.

Input

The input consists of a number of games. The first line of the input file specifies the number of games to follow. Each game contains the batting order of the visiting team followed by the batting order of the home team. The batting order of each team starts with a line specifying the team name (at most 15 upper and lower case letters). The next 9 lines specify the 9 players, listed in the order they bat. Each of these lines contains the player's name (at most 15 upper and lower case characters), a space, then a floating−point number (to 3 decimal places) specifying his hit percentage, a space, and finally his sacrifice percentage (to 3 decimal places). You may assume that all hit percentages are between .200 and .400, and all sacrifice percentages are between .300 and .750. You may assume that no game will last more than 200 innings.

Output

For each game, print the game number as well as the visiting and home team names in the first line, as follows:

```
Game <x>: <visiting> vs. <home>
```

This is followed by a blank line.

Next print the players who score hits and runs, separately, for each inning in the order they occur in the game. Use the format as follows:

```
Inning 1:
Hits:
  <player1> <team>
  <player2> <team>

Runs:
  <player3> <team>
  <player4> <team>
```

Indent the list of players by two spaces. Print the player name and the team name right−justified in a field width of 15 (in addition to the indentation before the player's name and one space separation between the player's name and team name). If no one scores a hit or a run, print the single line

```
  none
```

(indented by two spaces) in the appropriate section instead of a list of players. Print a blank line after the output for each inning.

At the end of the game, print a summary on the number of runs and hits scored for each team, starting with the visiting team:

```
End of Game:
  <visiting> <x> runs, <x> hits
```

```
  <home> <x> runs, <x> hits
```

The summary should be indented by two spaces. Print the team name right−justified in a field width of 15 (in addition to the indentation).

Separate the output of consecutive games by a line containing 60 '=' signs.

Sample Input

```
1
Rangers
Young .213 .523
Kinsler .207 .602
Sosa .254 .300
Laird .220 .432
Byrd .206 .749
Wilkerson .236 .508
Catalanotto .272 .483
Teixeira .297 .573
Saltalamacchia .243 .632
BlueJays
Wells .378 .502
Hill .276 .544
Overbay .372 .694
McDonald .373 .618
Adams .320 .690
Rios .300 .450
Johnson .379 .559
Stairs .302 .621
Zaun .346 .515
```

Sample Output

```
Game 1: Rangers vs. BlueJays

Inning 1:
Hits:
             Hill        BlueJays
          Overbay        BlueJays
            Adams        BlueJays

Runs:
  none

Inning 2:
Hits:
  none

Runs:
  none

Inning 3:
Hits:
      Catalanotto         Rangers
            Wells        BlueJays
             Hill        BlueJays
            Adams        BlueJays
```

```
Runs:
            Wells        BlueJays

Inning 4:
Hits:
          Kinsler         Rangers

Runs:
  none

Inning 5:
Hits:
       Catalanotto         Rangers

Runs:
  none

Inning 6:
Hits:
             Sosa         Rangers
            Laird         Rangers
             Rios        BlueJays

Runs:
  none

Inning 7:
Hits:
         Wilkerson         Rangers
          Teixeira         Rangers
            Stairs        BlueJays

Runs:
  none

Inning 8:
Hits:
  none

Runs:
  none

Inning 9:
Hits:
  none

Runs:
  none

End of Game:
           Rangers 0 runs, 7 hits
          BlueJays 1 runs, 8 hits
```