EuroPython 2013

Questions asked

Submissions

Moist

4pt	Not attempted **31/48 users** correct (65%)
6pt	Not attempted **31/31 users** correct (100%)

Captain Hammer

22pt	Not attempted **13/18 users** correct (72%)

Bad Horse

12pt	Not attempted **15/20 users** correct (75%)
21pt	Not attempted **13/15 users** correct (87%)

Professor Normal

12pt	Not attempted **2/5 users** correct (40%)
23pt	Not attempted **1/2 users** correct (50%)

Top Scores

6502	68
mgedmin	65
florentxicluna	65
koniiiik	65
kurazu	65
pts	65
fox91	65
harutune	65
mgax	44
spyyy	43

Problem A. Moist

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 1 4 points	Solve A-small-1
Small input 2 6 points	Solve A-small-2

Problem

Moist has a hobby -- collecting figure skating trading cards. His card collection has been growing, and it is now too large to keep in one disorganized pile. Moist needs to sort the cards in alphabetical order, so that he can find the cards that he wants on short notice whenever it is necessary.

The problem is -- Moist can't actually pick up the cards because they keep sliding out his hands, and the sweat causes permanent damage. Some of the cards are rather expensive, mind you. To facilitate the sorting, Moist has convinced Dr. Horrible to build him a sorting robot. However, in his rather horrible style, Dr. Horrible has decided to make the sorting robot charge Moist a fee of $1 whenever it has to move a trading card during the sorting process.

Moist has figured out that the robot's sorting mechanism is very primitive. It scans the deck of cards from top to bottom. Whenever it finds a card that is lexicographically smaller than the previous card, it moves that card to its correct place in the stack above. This operation costs $1, and the robot resumes scanning down towards the bottom of the deck, moving cards one by one until the entire deck is sorted in lexicographical order from top to bottom.

As wet luck would have it, Moist is almost broke, but keeping his trading cards in order is the only remaining joy in his miserable life. He needs to know how much it would cost him to use the robot to sort his deck of cards.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each one starts with a line containing a single integer, **N**. The next **N** lines each contain the name of a figure skater, in order from the top of the deck to the bottom.

Output

For each test case, output one line containing "Case #x: y", where x is the case number (starting from 1) and y is the number of dollars it would cost Moist to use the robot to sort his deck of trading cards.

Limits

1 ≤ **T** ≤ 100.
Each name will consist of only letters and the space character.
Each name will contain at most 100 characters.
No name with start or end with a space.
No name will appear more than once in the same test case.
Lexicographically, the space character comes first, then come the upper case letters, then the lower case letters.

Small dataset

1 ≤ **N** ≤ 10.

Large dataset

1 ≤ **N** ≤ 100.

Sample

```
Input                 Output

2                     Case #1: 1
2                     Case #2: 0
Oksana Baiul
Michelle Kwan
3
Elvis Stojko
Evgeni Plushenko
Kristi Yamaguchi
```

EuroPython 2013

Submissions

Moist

4pt	Not attempted **31/48 users** correct (65%)
6pt	Not attempted **31/31 users** correct (100%)

Captain Hammer

22pt	Not attempted **13/18 users** correct (72%)

Bad Horse

12pt	Not attempted **15/20 users** correct (75%)
21pt	Not attempted **13/15 users** correct (87%)

Professor Normal

12pt	Not attempted **2/5 users** correct (40%)
23pt	Not attempted **1/2 users** correct (50%)

Top Scores

6502	68
mgedmin	65
florentxicluna	65
koniiiik	65
kurazu	65
pts	65
fox91	65
harutune	65
mgax	44
spyyy	43

Problem B. Captain Hammer

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 22 points	Solve B-small

Problem

The Hamjet is a true marvel of aircraft engineering. It is a jet airplane with a single engine so powerful that it burns all of its fuel instantly during takeoff. The Hamjet doesn't have any wings because who needs them when the fuselage is made of a special Wonderflonium isotope that makes it impervious to harm.

Piloting the Hamjet is a not a job for your typical, meek-bodied superhero. That's why the Hamjet belongs to Captain Hammer, who is himself impervious to harm. The G-forces that the pilot endures when taking a trip in the Hamjet are legen-dary.

The Hamjet takes off at an angle of θ degrees up and a speed of **V** meters per second. **V** is a fixed value that is determined by the awesome power of the Hamjet engine and the capacity of its fuel tank. The destination is **D** meters away. Your job is to program the Hamjet's computer to calculate θ given **V** and **D**.

Fortunately, the Hamjet's Wondeflonium hull is impervious to air friction. Even more fortunately, the Hamjet doesn't fly too far or too high, so you can assume that the Earth is flat, and that the acceleration due to gravity is a constant 9.8 m/s^2 down.

Input

The first line of the input gives the number of test cases, **T**. **T** lines follow. Each line will contain two positive integers -- **V** and **D**.

Output

For each test case, output one line containing "Case #x: θ", where x is the case number (starting from 1) and θ is in degrees up from the the horizontal. If there are several possible answers, output the smallest positive one.

An answer will be considered correct if it is within 10^{-6} of the exact answer, in absolute or relative error. See the FAQ for an explanation of what that means, and what formats of floating-point numbers we accept.

Limits

1 ≤ **T** ≤ 4500;
1 ≤ **V** ≤ 300;
1 ≤ **D** ≤ 10000;
It is guaranteed that each test case will be solvable.

Sample

```
Input        Output

3            Case #1: 45.0000000
98 980       Case #2: 15.0000000
98 490       Case #3: 3.8870928
299 1234
```

Submissions

Moist

4pt	Not attempted **31/48 users** correct (65%)
6pt	Not attempted **31/31 users** correct (100%)

Captain Hammer

22pt	Not attempted **13/18 users** correct (72%)

Bad Horse

12pt	Not attempted **15/20 users** correct (75%)
21pt	Not attempted **13/15 users** correct (87%)

Professor Normal

12pt	Not attempted **2/5 users** correct (40%)
23pt	Not attempted **1/2 users** correct (50%)

Top Scores

6502	68
mgedmin	65
florentxicluna	65
koniiiik	65
kurazu	65
pts	65
fox91	65
harutune	65
mgax	44
spyyy	43

Problem C. **Bad Horse**

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 1 12 points	Solve C-small-1
Small input 2 21 points	Solve C-small-2

Problem

As the leader of the Evil League of Evil, Bad Horse has a lot of problems to deal with. Most recently, there have been far too many arguments and far too much backstabbing in the League, so much so that Bad Horse has decided to split the league into two departments in order to separate troublesome members. Being the Thoroughbred of Sin, Bad Horse isn't about to spend his valuable time figuring out how to split the League members by himself. That what he's got you -- his loyal henchman -- for.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case starts with a positive integer **M** on a line by itself -- the number of troublesome pairs of League members. The next **M** lines each contain a pair of names, separated by a single space.

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 "Yes" or "No", depending on whether the League members mentioned in the input can be split into two groups with neither of the groups containing a troublesome pair.

Limits

1 ≤ **T** ≤ 100.
Each member name will consist of only letters and the underscore character.
Names are case-sensitive.
No pair will appear more than once in the same test case.
Each pair will contain two distinct League members.

Small dataset

1 ≤ **M** ≤ 10.

Large dataset

1 ≤ **M** ≤ 100.

Sample

Input

```
2
1
Dead_Bowie Fake_Thomas_Jefferson
3
Dead_Bowie Fake_Thomas_Jefferson
Fake_Thomas_Jefferson Fury_Leika
Fury_Leika Dead_Bowie
```

Output

```
Case #1: Yes
Case #2: No
```

EuroPython 2013

A. Moist

B. Captain Hammer

C. Bad Horse

D. Professor Normal

Questions asked

Submissions

Moist

4pt	Not attempted **31/48 users** correct (65%)
6pt	Not attempted **31/31 users** correct (100%)

Captain Hammer

22pt	Not attempted **13/18 users** correct (72%)

Bad Horse

12pt	Not attempted **15/20 users** correct (75%)
21pt	Not attempted **13/15 users** correct (87%)

Professor Normal

12pt	Not attempted **2/5 users** correct (40%)
23pt	Not attempted **1/2 users** correct (50%)

Top Scores

6502	68
mgedmin	65
florentxicluna	65
koniiiik	65
kurazu	65
pts	65
fox91	65
harutune	65
mgax	44
spyyy	43

Problem D. Professor Normal

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 1 12 points	Solve D-small-1
Small input 2 23 points	Solve D-small-2

Problem

Professor Normal's three children have been causing trouble at school; so much so that the principal is threatening to expel them. Clearly, Professor Normal has been doing a great job raising the next generation of recruits for the Evil League of Evil. In order to appease the principal, he has agreed to organize a game for all of the students at the school, to teach them valuable life skills. After all, he can't afford to stay at home with the kids -- he has a job to do and a mortgage to pay.

The game is called *Don't Lose Your Marbles*. The children stand in an **M**-by-**N** grid. Each child is given some number of marbles to start. Different children may be given a different number of marbles; this teaches them that life is unfair.

The game then proceeds in turns. On each turn, each child gives out 12 marbles shared equally among his neighbours. *Neighbours* are the 4 children immediately in front, behind, to the left, and to the right of the child. Some children will have fewer than 4 neighbours, if they happen to be standing on the edges or in the corners of the grid. This exchange teaches the children about sharing.

Any children who do not have at least 12 marbles at the start of a turn are removed from the game before the marble exchange takes place. Their spots will then remain empty until the end of the game, and their neighbours will have fewer neighbours to exchange marbles with. This teaches the children that one must pay to play.

Children who do not have any neighbours are also then removed because they would have no one to exchange marbles with. This teaches them not to be selfish.

At this point, if there are no children left in the game, the game ends. Otherwise, every remaining child has somebody to exchange marbles with, and the game continues. This teaches the children that all things come to an end.

Or do they? Given the initial arrangement of the children and the number of marbles each child has, can you determine how many turns (marble exchanges) will take place? And please do so quickly, if you can. Professor Normal has evil to plan.

More precisely, the game proceeds according to the following algorithm:

```
number_of_exchanges = 0
repeat forever {
  while (any child has < 12 marbles or 0 neighbors) {
    remove all such children from play
  }
  if (there are no children left in play) {
    the game ends
  }
  simultaneously for each child in play {
    the child shares 12 marbles equally among neighbors
  }
  increment number_of_exchanges by 1
}
```

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each one starts with a line containing **M** followed by a line containing **N**. The next **M** lines each contain **N** space-separated integers -- the number of marbles for each child in that row of the grid.

Output

For each test case, output one line containing "Case #x: y turns", where x is the case number (starting from 1) and y is the number of marble exchanges that will take place before the game is over. If the game goes on forever, then instead output a line containing "Case #x: z children will play forever", where z is the number of children who will remain standing in the school's courtyard forever, playing the game.

Limits

1 ≤ **T** ≤ 100;

Small dataset

1 ≤ **M** ≤ 10;
1 ≤ **N** ≤ 10;
Each child will start with at most 100 marbles.

Large dataset

1 ≤ **M** ≤ 40;
1 ≤ **N** ≤ 40;
Each child will start with at most 10^{12} marbles.

Sample

```
Input        Output

6            Case #1: 2 turns
3            Case #2: 2 children will play forever
3            Case #3: 4 children will play forever
12 12 12     Case #4: 2 turns
12 12 12     Case #5: 0 turns
12 12 12     Case #6: 1 turns
2
3
12 12 12
12 12 12
2
2
12 13
14 15
2
3
11 12 11
20 14 20
1
1
74
2
2
10 12
12 12
```