ACM International Collegiate Programming Contest
2003 East Central Regional Practice Contest
Ashland University
Carnegie Mellon University
Sheridan University
University of Cincinnati
November 7, 2003

Sponsored by IBM

Rules:

1. There are **three** questions to be completed in **one hour and 30 minutes**.
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 PC^2 environment.
8. Judges' decisions are to be considered final. No cheating will be tolerated.

Problem A: Euchre Results

Anna, Betty, Cindy and Zelda like playing the card game Euchre. Euchre is a game for two teams of two, and each time they meet the girls split off into different teams. They also keep overall records of the number of games each player has won and lost. Anna has misplaced her won-loss results, but she does have the results of the other three players. Given this, she figures she can determine her won-loss record

Input

Input will consist of multiple problem instances. Each instance will consist of a single line containing six integers. The first two are the number of wins and losses (respectively) for Betty, the next two are the number of wins and losses for Cindy and the last two are the number of wins and losses for Zelda. A final line of six zeroes will terminate input and should not be processed.

Output

For each problem instance, output a single line indicating Anna's won-loss record, in the format shown in the example below.

Sample Input

```
10 3 6 7 8 5
1874 2945 2030 2789 1025 3794
0 0 0 0 0 0
```

Sample Output

```
Anna's won-loss record is 2-11.
Anna's won-loss record is 4709-110.
```

Problem B: Least Common Multiple

The *least common multiple (LCM)* of a set of positive integers is the smallest positive integer which is divisible by all the numbers in the set. For example, the LCM of 5, 7 and 15 is 105.

Input

Input will consist of multiple problem instances. The first line of the input file will contain a single integer indicating the number of problem instances. Each instance will consist of a single line of the form $m\ n_1\ n_2\ n_3 \ldots n_m$ where m is the number of integers in the set and $n_1 \ldots n_m$ are the integers. All integers will be positive and lie within the range of a 32-bit integer.

Output

For each problem instance, output a single line containing the corresponding LCM. All results will lie in the range of a 32-bit integer.

Sample Input

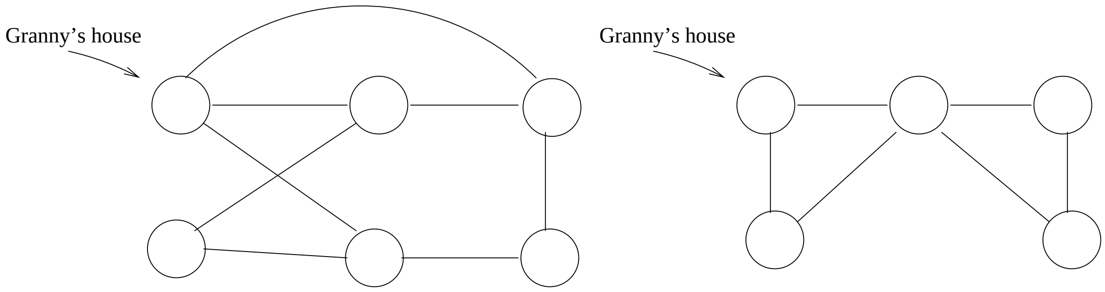
```
2
3 5 7 15
6 4 10296 936 1287 792 1
```

Sample Output

```
105
10296
```

Problem C: Granny's Bike

Most days Granny rides her bike around town to do errands, visit, have a cup of coffee, and so on. She enjoys riding her bike and wants to avoid passing the same place twice to add to the interest of the ride. So, each day she draws a map of the places to be visited, with lines connecting those near each other, and sees if she can visit them all and return home without passing a place more than once. Some days she finds she can do this and other days she finds she can't. For example, for the map on the left, Granny can visit every place and return home without passing any place twice, but she can't do it for the map on the right.



She turns to you to write a program to help her.

Input

There will be multiple test cases for this problem. Each test case will have input on multiple lines. The first line will contain the integer n (< 10) noting the number of places Granny wants to visit that day. These will be numbered 1 through n and Granny's house will be numbered 0. The next n lines will be a list of those places near each spot. The first line will be a list of places with a direct route from place 1. The second line will be a list of places with a direct route from place 2, and so on. You may assume that if place i has a direct route to place j, then there is a direct route the other direction also. A line containing 0 will follow the last test case.

Output

For each test case, print one line of the form:

`Case` m`: Granny can make the circuit.`

or

`Case` m`: Granny can not make the circuit.`

as appropriate. Here, m is the number of the test case, starting at 1.

Sample Input

```
5
0 2 5
0 1 3
2 4
0 3 5
1 4
4
0 2 3 4
1 3
1 2
0 1
0
```

Sample Output

```
Case 1: Granny can make the circuit.
Case 2: Granny can not make the circuit.
```