The CENTRE for EDUCATION
in MATHEMATICS and COMPUTING

2014 Canadian Computing Competition: Junior Division

Sponsor:

WATERLOO
MATHEMATICS

Canadian Computing Competition
Student Instructions for the Junior Problems

1. You may only compete in one competition. If you wish to write the Senior paper, see the other problem set.
2. Be sure to indicate on your **Student Information Form** that you are competing in the **Junior** competition.
3. You have three (3) hours to complete this competition.
   - if your supervising teacher is grading your solutions, all input is from the keyboard;
   - if you are using the On-line CCC grader, all input is from standard input;
   - all output is to standard output (i.e., to the screen).

   There is no need for prompting. Be sure your output matches the expected output in terms of order, spacing, etc. IT MUST MATCH EXACTLY!
4. Do your own work. Cheating will be dealt with harshly.
5. Do not use any features that the judge (your teacher or the On-line Grader) will not be able to use while evaluating your programs. In particular, take note of the type and version of the compiler used for your programming language on the On-line Grader if you are using the On-line Grader.
6. Books and written materials are allowed. Any machine-readable materials (like other programs which you have written) are *not* allowed. However, you are allowed to use "standard" libraries for your programming languages; for example, the STL for C++, java.util.*, java.io.*, etc. for Java, and so on.
7. Applications other than editors, compilers, debuggers or other standard programming tools are **not** allowed. Any use of other applications will lead to disqualification.
8. If your teacher is grading, please use file names that are unique to each problem: use `j1.pas` or `j1.c` or `j1.java` (or some other appropriate extension) for Problem J1. If you are using the On-line Grader, follow naming rules described there (and take particular note of file and class names for Java programs).
9. Your program will be run against test cases other than the sample ones. Be sure you test your program on other test cases. Inefficient solutions may lose marks for some problems. Be sure your code is as efficient (in terms of time) as possible. You will have at most 5 seconds of execution time per test case.
10. Check the CCC website at the end of March to see how you did on this contest and to see who the prize winners are. The CCC website is:

    `www.cemc.uwaterloo.ca/ccc`

Problem J1: Triangle Times

Problem Description

You have trouble remembering which type of triangle is which. You write a program to help.

Your program reads in three angles (in degrees).

- If all three angles are 60, output `Equilateral`.
- If the three angles add up to 180 and exactly two of the angles are the same, output `Isosceles`.
- If the three angles add up to 180 and no two angles are the same, output `Scalene`.
- If the three angles do not add up to 180, output `Error`.

Input Specification

The input consists of three integers, each on a separate line.

Each integer will be greater than 0 and less than 180.

Output Specification

Exactly one of `Equilateral`, `Isosceles`, `Scalene` or `Error` will be printed on one line.

Sample Input 1

```
60
70
50
```

Output for Sample Input 1

```
Scalene
```

Sample Input 2

```
60
75
55
```

Output for Sample Input 2

```
Error
```

Problem J2: Vote Count

Problem Description
A vote is held after singer A and singer B compete in the final round of a singing competition.

Your job is to count the votes and determine the outcome.

Input Specification
The input will be two lines. The first line will contain V $(1 \leq V \leq 15)$, the total number of votes. The second line of input will be a sequence of V characters, each of which will be A or B, representing the votes for a particular singer.

Output Specification
The output will be one of three possibilities:

- `A`, if there are more A votes than B votes;
- `B`, if there are more B votes than A votes;
- `Tie`, if there are an equal number of A votes and B votes.

Sample Input 1
```
6
ABBABB
```

Output for Sample Input 1
```
B
```

Sample Input 2
```
6
ABBABA
```

Output for Sample Input 2
```
Tie
```

Problem J3: Double Dice

Problem Description
Antonia and David are playing a game.

Each player starts with 100 points.

The game uses standard six-sided dice and is played in rounds. During one round, each player rolls one die. The player with the lower roll loses the number of points shown on the higher die. If both players roll the same number, no points are lost by either player.

Write a program to determine the final scores.

Input Specification
The first line of input contains the integer n $(1 \leq n \leq 15)$, which is the number of rounds that will be played. On each of the next n lines, will be two integers: the roll of Antonia for that round, followed by a space, followed by the roll of David for that round. Each roll will be an integer between 1 and 6 (inclusive).

Output Specification
The output will consist of two lines. On the first line, output the number of points that Antonia has after all rounds have been played. On the second line, output the number of points that David has after all rounds have been played.

Sample Input
```
4
5 6
6 6
4 3
5 2
```

Output for Sample Input
```
94
91
```

Explanation of Output for Sample Input
After the first round, David wins, so Antonia loses 6 points. After the second round, there is a tie and no points are lost. After the third round, Antonia wins, so David loses 4 points. After the fourth round, Antonia wins, so David loses 5 points. In total, Antonia has lost 6 points and David has lost 9 points.

Problem J4: Party Invitation

Problem Description

You are hosting a party and do not have room to invite all of your friends. You use the following unemotional mathematical method to determine which friends to invite.

Number your friends $1, 2, \ldots, K$ and place them in a list in this order. Then perform m rounds. In each round, use a number to determine which friends to remove from the ordered list.

The rounds will use numbers $r_1, r_2, \ldots, r_m$. In round i remove all the remaining people in positions that are multiples of r_i (that is, $r_i, 2r_i, 3r_i, \ldots$) The beginning of the list is position 1.

Output the numbers of the friends that remain after this removal process.

Input Specification

The first line of input contains the integer K $(1 \leq K \leq 100)$. The second line of input contains the integer m $(1 \leq m \leq 10)$, which is the number of rounds of removal. The next m lines each contain one integer. The ith of these lines $(1 \leq i \leq m)$ contains r_i ($2 \leq r_i \leq 100$) indicating that every person at a position which is multiple of r_i should be removed.

Output Specification

The output is the integers assigned to friends who were not removed. One integer is printed per line in increasing sorted order.

Sample Input

```
10
2
2
3
```

Output for Sample Input

```
1
3
7
9
```

Explanation of Output for Sample Input

Initially, our list of invitees is $1, 2, 3, 4, 5, 6, 7, 8, 9, 10$. There will be two rounds of removals. After the first round of removals, we remove the even positions (i.e., every second position), which causes our list of invitees to be $1, 3, 5, 7, 9$. After the second round of removals, we remove every 3rd remaining invitee: thus, we keep 1 and 3, remove 5 and keep 7 and 9, which leaves us with an invitee list of $1, 3, 7, 9$.

Problem J5: Assigning Partners

Problem Description
The CEMC is organizing a workshop with an activity involving pairs of students. They decided to assign partners ahead of time. You need to determine if they did this *consistently*. That is, whenever A is a partner of B, then B is also a partner of A, and no one is a partner of themselves.

Input Specification
The input consists of three lines. The first line consists of an integer N $(1 < N \leq 30)$, which is the number of students in the class. The second line contains the first names of the N students separated by single spaces. (Names contain only uppercase or lowercase letters, and no two students have the same first name). The third line contains the same N names in some order, separated by single spaces.

The positions of the names in the last two lines indicate the assignment of partners: the ith name on the second line is the assigned partner of the ith name on the third line.

Output Specification
The output will be `good` if the two lists of names are arranged consistently, and `bad` if the arrangement of partners is not consistent.

Sample Input 1
```
4
Ada Alan Grace John
John Grace Alan Ada
```

Output for Sample Input 1
```
good
```

Explanation for Output for Sample Input 1
Ada and John are partners, and Alan and Grace are partners. This arrangement is consistent.

Sample Input 2
```
7
Rich Graeme Michelle Sandy Vlado Ron Jacob
Ron Vlado Sandy Michelle Rich Graeme Jacob
```

Output for Sample Input 2
```
bad
```

Explanation for Output for Sample Input 2
Graeme is partnered with Vlado, but Vlado is partnered with Rich. This is not consistent. It is also inconsistent because Jacob is partnered with himself.