The CENTRE for EDUCATION
in MATHEMATICS and COMPUTING

2014 Canadian Computing Competition: Senior Division

Sponsor:

WATERLOO
MATHEMATICS

Canadian Computing Competition
Student Instructions for the Senior Problems

1. You may only compete in one competition. If you wish to write the Junior paper, see the other problem set.

2. Be sure to indicate on your **Student Information Form** that you are competing in the **Senior** competition.

3. You have three (3) hours to complete this competition.

4. You should assume that:

   - if your supervising teacher is grading your solutions, all input is from a file named `sX.in`, where `X` is the problem number ($1 \leq \texttt{X} \leq 5$);
   - 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!

5. Do your own work. Cheating will be dealt with harshly.

6. 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.

7. 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.

8. Applications other than editors, compilers, debuggers or other standard programming tools are **not** allowed. Any use of other applications will lead to disqualification.

9. If your teacher is grading, please use file names that are unique to each problem: use `s1.pas` or `s1.c` or `s1.java` (or some other appropriate extension) for Problem S1. 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).

10. 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, especially Problems 4 and 5.

11. The Senior problems shall be given 5 seconds of execution time per test case (if graded in schools), and similarly, 5 seconds of execution time on the on-line grader (if graded using the on-line grader). If this time limit is exceeded, no points will be awarded for that test case.

12. If you finish in the top 20 competitors on this competition, you will be invited to participate in CCC Stage 2, held at the University of Waterloo in May 2014. We will select the Canadian International Olympiad in Informatics (IOI) team from among the top contestants at the CCC Stage 2. You should note that IOI 2014 will be held in Taiwan. Note that you will need to know C, C++ or Pascal if you are invited to the CCC Stage 2. But first, do well on this contest!

13. 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/contests/computing.html`

Problem S1: 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 S2: 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
```

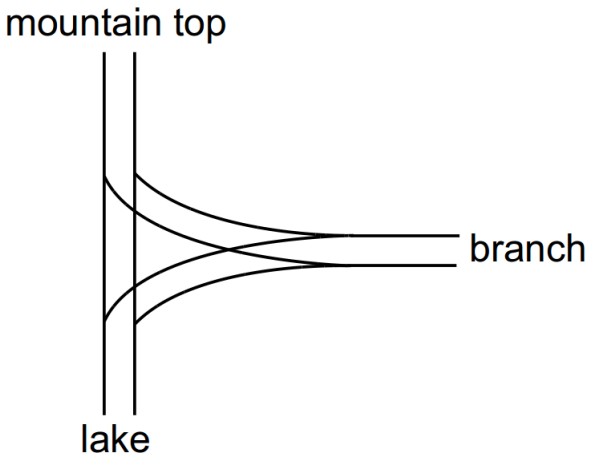
Output for Sample Input 2
```
bad
```

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.

Problem S3: The Geneva Confection

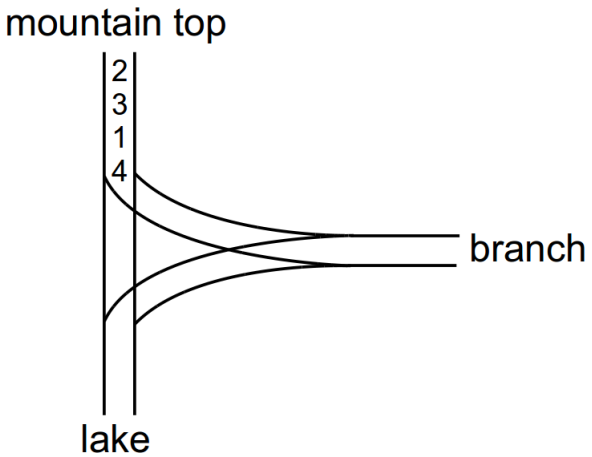
Problem Description

In order to ensure peace and prosperity for future generations, the United Nations is creating the world's largest candy. The ingredients must be taken in railway cars from the top of a mountain and poured into Lake Geneva. The railway system goes steeply from the mountaintop down to the lake, with a T-shaped branch in the middle as shown below.



Right now, each of the N ingredients is in its own railway car. Each railway car is assigned a positive integer from 1 to N. The ingredients must be poured into the lake in the order $1, 2, 3, \ldots, N$ but the railway cars are lined up in some random order. The difficulty is that, because of the especially heavy gravity today, you can only move cars downhill to the lake, or sideways on the branch line. Is it still possible to pour the ingredients into the lake in the order $1, 2, 3, \ldots, N$?

For example, if the cars were in the order $2, 3, 1, 4$, we can slide these into the lake in order as described below:



- Slide car 4 out to the branch
- Slide car 1 into the lake
- Slide car 3 out to the branch
- Slide car 2 into the lake
- Slide car 3 from the branch into the lake
- Slide car 4 from the branch into the lake

Input Specification

The first line will contain the number T ($1 \leq T \leq 10$) which is the number of different tests that will be run. Each test has the form of an integer N ($1 \leq N \leq 100\,000$) on the first line of the test, followed by a list of the N cars listed from top to bottom. The cars will always use the numbers from 1 to N in some order.

Output Specification

For each test, output one line which will contain either `Y` (for "yum") if the recipe can be completed, and `N` otherwise.

Sample Input

```
2
4
2
3
1
4
4
4
1
3
2
```

Output for Sample Input

```
Y
N
```

Problem S4: Tinted Glass Window

Problem Description
You are laying N rectangular pieces of grey-tinted glass to make a stained glass window. Each piece of glass adds an integer value "tint-factor". Where two pieces of glass overlap, the tint-factor is the sum of their tint-factors.

You know the desired position for each piece of glass and these pieces of glass are placed such that the sides of each rectangle are parallel to either the x-axis or the y-axis (that is, there are no "diagonal" pieces of glass).

You would like to know the total area of the finished stained glass window with a tint-factor of at least T.

Input Specification
The first line of input is the integer N ($1 \leq N \leq 1000$), the number of pieces of glass. The second line of input is the integer T ($1 \leq T \leq 1\,000\,000\,000$), the threshold for the tint-factor. Each of the next N lines contain five integers, representing the position of the top-left and bottom-right corners of the ith piece of tinted glass followed by the tint-factor of that piece of glass. Specifically, the integers are placed in the order $x_l\ y_t\ x_r\ y_b\ t_i$, where the top-left corner is at (x_l, y_t) and the bottom-right corner is at (x_r, y_b), and tint-factor is t_i. You can assume that $1 \leq t_i \leq 1\,000\,000$. The top-most, left-most co-ordinate where glass can be placed is $(0, 0)$ and you may assume $0 \leq x_l < x_r \leq K$ and $0 < y_t < y_b \leq K$, and

The following additional constraints will apply.

- At least 10% of the marks will be for test cases where $N \leq 100$ and $K \leq 100$;
- at least 30% of the marks will be for test cases where $N \leq 1000$ and $K \leq 1000$;
- at least 40% of the marks will be for test cases where $N \leq 100$ and $K \leq 1\,000\,000\,000$;
- the remaining marks will be for test cases where $N \leq 1000$ and $K \leq 1\,000\,000\,000$.

Output Specification
Output the total area of the finished stained glass window which has a tint-factor of at least T. All output will be less than 2^{64}, and the output for some test cases will be larger than 2^{32}.

Sample Input

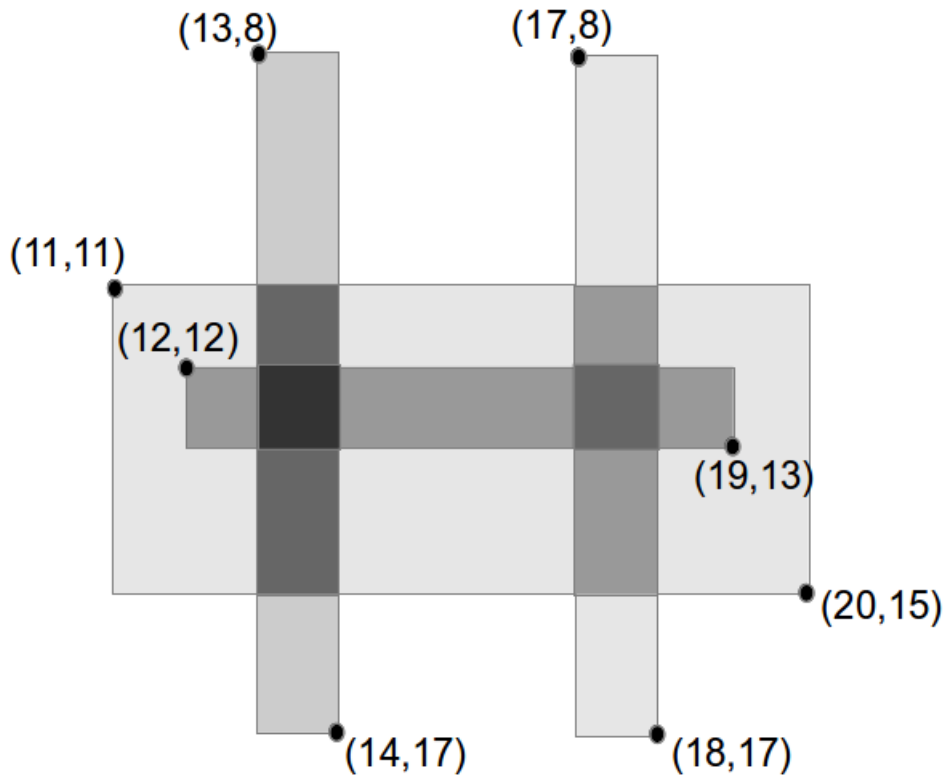
```
4
3
11 11 20 15 1
13 8 14 17 2
17 8 18 17 1
12 12 19 13 1
```

Output for Sample Input

```
5
```

Explanation of Output for Sample Input

There are 4 pieces of glass used. There are two regions of glass which have a tint-factor greater than or equal to 3: one region between $(13, 11)$ and $(14, 15)$ (which has tint-factor of 3, except for a unit square with tint-factor 4), and another region between $(17, 12)$ and $(18, 13)$ (with tint-factor 3). In total, these two regions have 5 square units of glass with tint-factor greater than or equal to 3, as shown on the diagram below.



Problem S5: Lazy Fox

Problem Description
You have a pet Fox who loves treats. You have N neighbours at distinct locations (described as points on the Cartesian plane) which hand out treats to your pet Fox, and each neighbour has an unlimited number of treats to give out. The origin (which is where the Fox starts) will not be one of these N locations.

What does the Fox say, in order to get these treats? That is a good question, but not our concern.

The Fox moves from location to location to gather exactly one treat from each location on each visit. He can revisit any previous location, but cannot visit the same location on two *consecutive* visits.

Your Fox is very lazy. The distance your Fox is willing to travel after each treat will strictly decrease. Specifically, the distance from the origin to his first treat location must be larger than the distance from his first treat location to his second treat location, which in turn is larger than the distance between his second treat location and his third treat location, and so on.

What is the largest number of treats your Fox gathers?

Input Specification
The first line contains the integer N ($1 \leq N \leq 2000$). The next N lines each contain X_i, followed by a space, followed by Y_i, for $i = 1..N$ ($-10\,000 \leq X_i, Y_i \leq 10\,000$) representing the coordinates of the ith location. The following additional constraints will apply.

- At least 20% of the marks will be for test cases where $N \leq 50$;
- at least 40% of the marks will be for test cases where $N \leq 200$;
- the remaining marks will be for test cases where $N \leq 2000$.

Output Specification
The output is one integer, the largest number of treats your Fox can gather.

Sample Input
```
5
5 8
4 10
3 1
3 2
3 3
```

Output for Sample Input
```
6
```

Explanation of Output for Sample Input
The Fox performs the visits in the following order (with the indicated distances):

- $(0, 0)$ to $(4, 10)$ with distance $\sqrt{116}$;
- $(4, 10)$ to $(3, 1)$ with distance $\sqrt{82}$;
- $(3, 1)$ to $(5, 8)$ with distance $\sqrt{53}$;
- $(5, 8)$ to $(3, 3)$ with distance $\sqrt{29}$;
- $(3, 3)$ to $(3, 1)$ with distance 2;
- $(3, 1)$ to $(3, 2)$ with distance 1.