VCU School of Engineering
2013 Computer Science Programming Contest

Welcome to the 2013 Programming Contest. Before you start the contest, please be aware of the following:

1. There are ten (10) problems in the packet, using letters A-J. These problems are NOT sorted by difficulty. When a team's solution is judged correct, the team will be awarded a balloon. The balloon colors are as follows:

Problem	Problem Name	Balloon Color
A	Card Shark	Dark Blue
B	Around and Around	Light Blue
C	Fence in the Battlefield	Gold
D	A New Lottery Game	Green
E	Government Inefficiency!	Ivory
F	Triangle, Triangle, Triangle	Purple
G	Historic Photos	Orange
H	Simply Fractions	Yellow
I	Wormy Wormholes	Black
J	Poogle Equality	Pink

2. All solutions must read from standard input (System.in) and write to standard output (System.out).

3. Solutions for problems submitted for judging are called runs. Each run will be judged. Runs for each particular problem will be judged in the order in which they are received. However, it is possible that runs for different problems may be judged out of order. For example, you may submit a run for B followed by a run for C, but receive the response for C first. DO NOT request clarifications on when a response will be returned. If you have not received a response for a run within 10 minutes of submitting it, **you may ask**

the lab monitor to send a runner to the judge to determine the cause of the delay. Under no circumstances should you ever submit a clarification request about a submission for which you have not received a judgment.

The judges will respond to your submission with one of the following responses. In the event that more than one response is applicable, the judges may respond with any of the applicable responses.

Response	Explanation
Yes	Your submission has been judged correct.
Wrong Answer	Your submission generated output that is not correct.
Output Format Error	Your submission's output is not in the correct format, is misspelled, or did not produce all of the required output.
Excessive Output	Your submission generated output in addition to what is required.
Compilation Error	Your submission failed to compile.
Run-Time Error	Your submission experienced a run-time error.
Time-Limit Exceeded	You submission did not solve the judges' test data within **2 minutes**.

4. A team's score is based on the number of problems they solve and penalty points. The penalty points reflect the time required to solve a problem correctly and the number of incorrect submissions made before the problem is solved. For each problem solved correctly, penalty points are charged equal to the time at which the problem was solved plus 20 minutes for each incorrect submission. No penalty points are added for problems that are never solved. Teams are ranked by the number of problems solved. Ties are resolved in favor of the team with the fewest penalty points.

5. This problem set contains sample input and output for each problem. However, you may be assured that the judges will test your submission against several other more complex datasets, which will not be revealed. Before submitting your run you should design other input sets to fully test your program. Should you receive an incorrect judgment, you are advised to consider what other datasets you could design to further evaluate your program.

6. In the event that you think a problem statement is ambiguous, you may request a clarification. Read the problem carefully before requesting a clarification. If the judges believe that the problem statement is sufficiently clear, you will receive the response,

"The problem statement is sufficient, no clarification is necessary." If you receive this response, you should read the problem description more carefully. If you still think there is an ambiguity, you will have to be more specific or descriptive of the ambiguity you have found. If the problem statement is ambiguous in specifying the correct output for a particular input, please include that input data in the clarification request.

Additionally, you may submit a clarification request asking for the correct output for input you provide. The judges will seek to respond to these requests with the correct output. These clarification requests will be answered only when no clarifications regarding ambiguity are pending. The judges reserve the right to suspend responding to these requests during the contest.
If a general clarification, including output for a given input, is issued during the contest, it will be broadcast to all teams.

7. The submission of abusive programs or clarification requests to the judges will be considered grounds for immediate disqualification.

8. Good luck and HAVE FUN!!

Problem A
Card Shark

A new card game is sweeping the nation. It has become the game of choice in Petersburg with weekly card games being held as part of the Friday at the Arts Celebration. This game is similar to Poker, but no attention is paid to suits, and we don't care about straights. The idea is that the person with the worst hand wins. Therefore you do not want any pairs.

David is a very conservative player. He only wants to bet if he is really sure that he will win. So you are to write a program to tell him if he has the worst hand possible. Each hand, seven cards are dealt to each player. They may then discard 2 cards. A guaranteed winning hand will have no pairs, and the lowest possible "high" card.

Input

The first line of input will be an integer telling you how many hands to evaluate. This is followed by one line for each hand. Each line will have 7 integers separated by spaces. We use 1 for an Ace, 11 for a Jack, 12 for a Queen and 13 for a King.

Output

For each line of input read, you are to output either "Bet" or "Fold" on a separate line. Only output "Bet" if you have the worst possible hand. Otherwise, output "Fold".

Sample Input

```
5
1 2 3 4 5 6 7
1 2 3 4 6 7 8
1 2 3 3 4 4 4
11 1 5 4 12 2 3
5 1 3 2 6 7 9
```

Sample Output 1

```
Bet
Fold
Fold
Bet
Fold
```

Problem B
Around and Around

The City of Treasures is laid out in a neat grid of intersecting streets. On each block lays a wondrous building that is an architectural delight. The buildings get more and more amazing as one reaches the center most block where the Palace of Philosophical Thought is found. Visitors are let off a various locations in the city and from there they spiral into the center. However, the streets have no names, and the buildings, except for the Palace have no names. Instead simple plaques are mounted outside each building door giving the number of the building. You are to write a program that will give the sequence in which buildings are to be visited in order to spiral into the center.

Visitors, when dropped off, are starting at the corner of a square. They then traverse the edges of the square, spiraling in as they complete each traversal of the square until they reach the center.

Input

The first line of input is a single integer, n, giving the length of the side of the outside square that they are to traverse. The next n lines then each contain n integers separated by spaces, giving the numbers of the buildings they will visit.

Output

Your program should print out the building numbers in the order that they will be visited. This list of numbers should be all on one line and the numbers should be separated by a single space

Sample Input 1

```
3
1 2 3
4 5 6
7 8 9
```

Sample Output 1

```
1 2 3 6 9 8 7 4 5
```

Sample Input 2

```
4
1 2 3 4
5 7 6 8
9 10 11 12
13 14 15 16
```

Sample Output 2

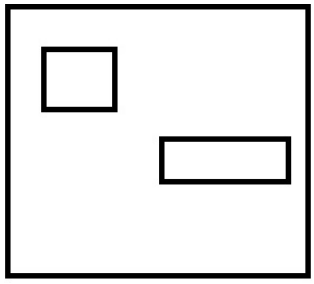
```
1 2 3 4 8 12 16 15 14 13 9 5 7 6 11 10
```

Problem C
Fence in the Battlefield

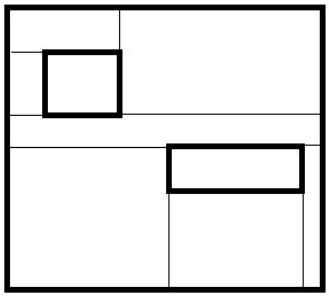
There are many historic Civil War battlefields near Petersburg, Virginia. However, this is a rural area, and one particular battlefield is badly overgrown. An enterprising gentleman has suggested fencing in the battlefield in plots that will each hold several cows. The cows will take care of the tall grass and we will have grass-fed beef to eat. The battlefield is 100 meters by 100 meters. (OK, so it was a small battle!)

On inspection, it was found that some areas of the battlefield have already been fenced in. None of these rectangles overlap – although they may touch each other. These are areas that contain gravesites or unsafe areas. Your goal is to find the minimum number of new rectangles needed to enclose the entire battlefield into rectangles that are adjacent to one another. All enclosed areas must be rectangles.

For example, below is a picture of the battlefield with 2 fenced in areas. The outside edges are not fences but are simply marked to illustrate the size of the field.



Now here is an illustration showing how to add fences (the thinner lines) to make the rectangles. This example added seven new rectangles.



Input

The first line contains an integer telling you the number of fenced in rectangles that already exist (max 10). On each of the following n lines are 4 integers separated by spaces. The first 2 give the x,y coordinates of one corner. The second 2 give the x,y coordinates of the opposite corner. All rectangles will be at least 3 meters wide in both directions.

Output
A single integer giving the minimum number of rectangles required.

Sample Input
2
10 10 40 40
60 50 75 90

Sample Output
7

Problem D
A New Lottery Game

An eccentric old philosopher and philanthropist has decided to give his money away using a game of his own devising. People buy a number and are placed in the lottery system in the order in which they purchase a ticket. People may choose what number they like, but their ordering in the system is given exactly by the time at which they buy the ticket. No two tickets can be sold at the same time, but many people may choose the same number. Ticket sales stop after 100,000 tickets have been sold.

Three winners (a winning set) are chosen from the list. Here are the criteria for being chosen: The first person in a winning set of three has both a lower number and bought their ticket before the second person in the winning set. The second person in the winning set has both a lower number and bought their ticket before the third person in the winning set.

For a given sale of tickets you must identify how many possible winning sets there are for that sale.

Input
The first line will tell you how many people bought tickets during this sale.
The second line will have one integer for each person, separated by spaces, in the order that they bought their ticket.

Output
A single integer telling how many winning sets of three exist for this ticket sale.

Sample Input

5
12 23 11 25 37

Sample Output

5

Problem E
Government Inefficiency!

The Assistant to the Assistant of the Advisor to the Cheese Oversight Board had been gradually hiring all her family to work for her. It reached the point where she had 427 underlings (including a second cousin twice removed by marriage!). Unable to stop this, the City Board passed a new law that any city employee could only have 2 people working directly for them. Further, they could only issue directives to their immediate subordinates. They should have restricted this new law to just the Cheese Oversight Board, but they didn't and so it affects all of the city government.

The side of effect of this is that instructions for workers now have to come down the chain of command. This seemed to be slowing down the workings of government, so the city rushed out and hired ACME Consulting to do a study of the situation. ACME reported that the average communication of a directive from employee to direct subordinate took an average of 14.35 minutes (that is, 14 minutes, 21 seconds).

Now, we can't fire government employees, so they have hired you to find a way of restructuring the chain of communication so that the longest time from Mayor to lowly worker is minimized. This means that if there are 10 employees, you can structure the chain of communication in any way you choose as long as each person passes on the message to no more than 2 others. An employee may tell both of their subordinates at the same time if they choose to do so.

Input

There will be a single integer input which will give the number of lines of input to follow. Each line of input after that will contain a single integer giving the number of employees the city has (no one seems to agree on this number!)

Output

For each line of input you are to output the average amount of time it would take for communications to get from the Mayor to the worker furthest away in the communication chain. Give this answer rounded to the nearest whole number of minutes.

Sample Input

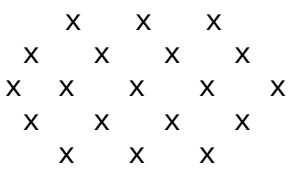
```
2
7
255
```

Sample Output

```
29
100
```

Problem F
Triangle, Triangle, Triangle

School children were asked to decorate the sides of old barns with hexagons of stars. They liked this – they got to get out of the school building and spend the afternoon in the country painting. Here is an example of what one of them painted:



This is a hexagon of length 2 on each side. The next day the teacher took them back out to see their artwork, and (horror of horrors) set them the following math question! For a given hexagon of length n on each side, how many equilateral triangles can you find where each triangle side is also of length n? A straight line from 1 star to the next is considered length 1 – all triangle sides of length n must start at a star, end at a star, and pass through n-1 intermediate stars.

Input
A single integer telling you the length of the side of the hexagon.

Output
Print out a single value giving the number of triangles.

Sample Input

1

Sample Output

6

Problem G
Historic Photos

Historic Petersburg Foundation is designing a photo display that will showcase the beautiful old buildings in the city. They have hired a firm to design the photo layout for them. The firm has sent them a file describing the layout by giving the location of each photo on 100 inch square board. However, no one remembered to tell the firm that none of the photos should overlap. You are to write a program to find out if any of them do overlap.

Input

The first line will consist of a single integer n which will give the number of datasets. The following lines will describe the datasets. The first line of each dataset will consist of a single integer m which will give the number of photos to be displayed for the dataset. The next m lines of the dataset will each contain 4 integers separated by spaces. The first two numbers will give the (x,y) coordinates of one corner of the photo. The second two numbers will give the (x,y) coordinates of the opposite corner of the photo.

Output

A series of n lines, each line will read either “Overlap” or “No Overlap”

Sample Input

```
2
3
1 1 2 2
2 5 4 7
4 2 7 3
2
1 1 3 3
2 2 5 3
```

Sample Output

```
No Overlap
Overlap
```

Problem H
Simply Fractions

Simple Simon is a pie man who wants simple fractions. He can cut his pies into ½ or 1/3 or 1/10 or 1/659. He doesn't like fractions with numerators bigger than 1. So when a customer asks for 2/17 of a cherry pie he is unhappy. Of course, he could simply cut 1/17 of the pie twice, but that would be boring. He likes cutting different fractions. You are to write a program that will take a fraction and convert it to a sum of fractions that all have the numerator 1 and all have different denominators. For example:

3/5 = ½ + 1/10

Input

The first line of input will contain two integers separated by spaces. The first is the numerator and the second is the denominator. The second line will give you a second fraction in the same format.

Output

You are to output just the denominator values for the fractions that add up to the input fraction. These values should be printed from smallest to largest, one per line. After the list of denominators is complete for the first fraction, you should print a line with "x-x-x-x-x-x" before outputting the second list of denominators.

Sample input

3 5
521 10250

Sample Output

2
10
x-x-x-x-x-x
20
1206
12361500

Problem I
Wormy Wormholes

Your spaceship fell into a time-space anomaly and you have ended up in Theta-Umber sector. Luckily your navigator realized where you were by recognizing the unusual purple haze atmosphere of the planet Arodz. With this information and your computers complete database of all star systems and wormholes you think you can get home. A wormhole is a two-way tunnel through space-time connecting two points in space, each in a different star system.

Each time you travel through a wormhole it takes a toll on your shields. The longer the wormhole, the bigger the toll. If your shields give way then you are toast. Chief Engineer Ghosh has reported that the shields were damaged when you fell through the anomaly. He recommends that you travel through no wormhole longer than 10 light years. You may consider the universe cut into star systems. Each star system has a corresponding ID. You may easily travel within a star system without any real wear and tear on your spaceship. You cannot travel to another star system without going through a wormhole.

So, can you get home? Or are you going to have to travel the universe looking for a space junkyard where you can get the pieces to repair your shield?

Input

The first line of input is a single integer n giving the number of wormholes in the universe that are of length less than 10 light years. The next n lines each consist of two integers separated by a space. The first integer is the ID of the star system where the wormhole starts. The second is the ID of the end of the wormhole.

The next line contains the number of problems to solve, m. Each of the following m lines contains two integers separated by a space, defining the problem. The first is the ID of the current star system you are in. The second is the ID of the star system you call home. You are to determine if home can be reached from the current position.

Output

For each of the m problems, you should print a single line. If you find a way home, then print out the line "ENGAGE WORMHOLE TRAVEL PLAN". If you do not find a way home, then print out the line "COMPUTER, LOCATE NEAREST JUNKYARD"

Sample Input

```
5
17 32
45 17
687 32
687 1
3 6
3
17 1
6 32
45 687
```

Sample Output

```
ENGAGE WORMHOLE TRAVEL PLAN
COMPUTER, LOCATE NEAREST JUNKYARD
ENGAGE WORMHOLE TRAVEL PLAN
```

Problem J
Poogle Equality

In the world of Similarity people are judged by the number of poogles that they own. The more poogles you have the lower your status. However, it is hard to get rid of poogles – no one will take them from you, and if you try to abandon them they just fly right back to you. When a person dies, their poogles become the property of their nearest blood relative.

In order to right this inequality, the government (who else would right an inequality?) has established the Ministry for Poogle Redistribution and Care. The Ministry has a poogle redistribution plan. If you apply you can participate in redistribution. However, the Ministry sometimes also forces people with just a few (or no) poogles to participate.

The Ministry takes people in groups of 4 for redistribution. The 4 people, call them A, B, C, and D form a circle around the minister in charge. The minister then calculates

|poogles A has – poogles B has|

and gives that number of poogles to A. This process is repeated for B and C, C and D, and D and A. This completes one cycle of redistribution. The cycles are repeated until everyone has the same number of poogles.

The Most High Minister is interested in the relationship between the number of poogles people have and the number of cycles needed to complete redistribution. You are to write a program that will, given the number of cycles, tell the Most High Minister the smallest maximum number of poogles that one person must have in order to need that number of cycles.

For example, consider having 3 cycles. Then you must have at least one person with 1 poogle. If you have A=1, B=0, C=0, D=0, then you will get:

after cycle 1: A=1, B=0,C=0, D=1
after cycle 2: A=1, B=0, C=1, D=0
after cycle 3: A=1, B=1, C=1, D=1

Input

There will be one line of input containing 1 integer giving the number of cycles.

Output

You should output a single integer giving the smallest maximum number of poogles that someone must have.

Sample Input 1

3

Sample Output 1

1

Sample Input 2

5

Sample Output 2

3