Nivel “Libres”
Problemset

October 14th 2016

Problem A: Goldbach Graphs

Christian Goldbach sent a letter to Leonhard Euler in 1742 in which he made the following conjecture:

"Every even number greater than 4 can be written as the sum of two odd prime numbers"

To find the solutions of Goldbach's conjecture for a given even number n ($n > 0$), let us define the directed graph GG(n) (the Goldbach Graph of n) as follows:

Nodes are prime numbers p such that $1 < p < n$.

For each node p there are zero or more outgoing edges, determined by the following rules:

- If $p + q = n$ and $q = 1$, then no outgoing edges are related to p.
- If $p + q = n$ and $q = p1\ p2\ p3\ ....\ pk$ is the prime factorization of q (assuming $q > 1$), then for each $i = 1..k$ an edge $p->pi$ is added to graph GG(n). Notice that each pi must be a prime number. Besides, if $k = 1$ then q is prime and we have a solution to Goldbach's conjecture.

For example:

- GG(2) is empty (it has zero nodes)
- GG(4) has two nodes and one edge.
  - nodes = {2, 3}
  - edges = {2->2}
- GG(6) has three nodes and three edges
  - nodes = {2, 3, 5}
  - edges = {2->2, 2->2, 3->3}
  - Notice that edge 2->2 appears twice in GG(6) because when $p = 2$ then $q = 4 = 2*2$

Solutions to Goldbach's conjecture are cycles in graph GG(n) of the following types:

- Single-node cycles (Type I): a node p with only one outgoing edge $p->p$.
- Double-node cycles (Type II): two nodes $p1$ and $p2$, such that each one has a unique outgoing edge ($p1->p2$, $p2->p1$).

Your task is to inspect the directed graph GG(n) starting from a given node x and searching every node reachable from x for a solution to Goldbach's conjecture. The procedure is successful if a node belonging to a Type I or Type II cycle is found. In such a case the minimum distance from x to the first node of the cycle found must be reported. Otherwise it should be stated that a solution can not be found.

Your algorithm should take into account that GG(n) can contain other types of cycles besides the ones described here. Otherwise, it can run forever.

Input

The input contains several lines each one with a different test case. Each line includes a pair of numbers representing the values *n* and *x*. You should assume that *n* is even and also that 2 <= *n* <= 1000. Although 0 < *x* < *n* is true, do not assume that *x* is a valid node of GG(*n*). The last line of the input contains the number 0 (it is not a test case).

Output

For each test case output a single line with one of the following:

- Solution found at distance *D*.
- Solution not reachable.
- *x* is not a node!

Where *D* is the minimum distance from *x* to the solution found, as described before.

Examples

Input	Output
2 1	1 is not a node!
4 2	Solution found at distance 0.
6 2	Solution not reachable.
6 3	Solution found at distance 0.
12 3	Solution not reachable.
12 11	Solution not reachable.
14 7	Solution found at distance 0.
20 5	Solution found at distance 1.
38 11	Solution found at distance 2.
50 17	Solution found at distance 1.
540 340	340 is not a node!
540 31	Solution found at distance 0.
540 33	33 is not a node!
0	

Problem B: Einbahnstrasse

Einbahnstrasse (German for a one-way street) is a street on which vehicles should only move in one direction. One reason for having one-way streets is to facilitate a smoother flow of traffic through crowded areas. This is useful in city centers, especially old cities like Cairo and Damascus. Careful planning guarantees that you can get to any location starting from any point. Nevertheless, drivers must carefully plan their route in order to avoid prolonging their trip due to one-way streets. Experienced drivers know that there are multiple paths to travel between any two locations. Not only that, there might be multiple roads between the same two locations. Knowing the shortest way between any two locations is a must! This is even more important when driving vehicles that are hard to maneuver (garbage trucks, towing trucks, etc.)

You just started a new job at a car-towing company. The company has a number of towing trucks parked at the company's garage. A tow-truck lifts the front or back wheels of a broken car in order to pull it straight back to the company's garage. You receive calls from various parts of the city about broken cars that need to be towed. The cars have to be towed in the same order as you receive the calls. Your job is to advise the tow-truck drivers regarding the shortest way in order to collect all broken cars back in to the company's garage. At the end of the day, you have to report to the management the total distance traveled by the trucks.

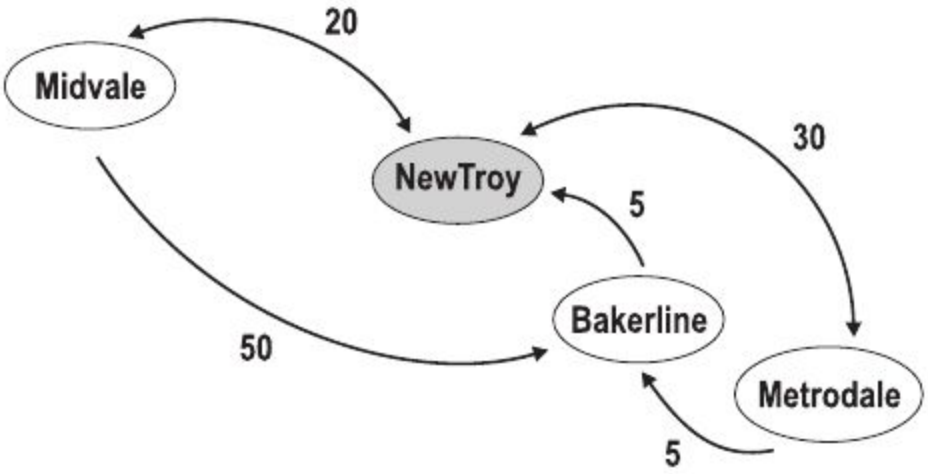
Input

Your program will be tested on one or more test cases. The first line of each test case specifies three integer numbers (**N** , **C** , and **R**) separated by one or more spaces. The city has **N** locations with distinct names, including the company's garage. **C** is the number of broken cars. **R** is the number of roads in the city. Note that 0 < **N** < 100 , 0 <= **C** < 1000 , and **R** < 10000 . The second line is made of C + 1 words, the first being the location of the company's garage, and the rest being the locations of the broken cars. A location is a word made of 10 letters or less. Letter case is significant. After the second line, there will be exactly **R** lines, each describing a road. A road is described using one of these three formats:

A --v-> B
A <-v-- B
A <-v-> B

A and B are names of two different locations, while v is a positive integer (not exceeding 1000) denoting the length of the road. The first format specifies a one-way street from location A to B , the second specifies a one-way street from B to A , while the last specifies a two-way street between them. A , "the arrow", and B are separated by one or more spaces. The end of the test cases is specified with a line having three zeros (for **N**, **C**, and **R**.)

The test case in the example below is the same as the one in the figure.



Output

For each test case, print the total distance traveled using the following format:

k. V

Where k is test case number (starting at 1,) is a space, and V is the result.

Example

Input	Output
4 2 5 NewTroy Midvale Metrodale NewTroy <-20-> Midvale Midvale --50-> Bakerline NewTroy <-5-- Bakerline Metrodale <-30-> NewTroy Metrodale --5-> Bakerline 0 0 0	1. 80

Problem C: Odd Occurrences in Array

A non-empty set A consisting of **N** integers is given. The set contains an odd number of elements, and each element of the set can be paired with another element that has the same value, except for one element that is left unpaired.
For example, in a set A such that:

```
A[0] = 9  A[1] = 3  A[2] = 9
A[3] = 3  A[4] = 9  A[5] = 7
A[6] = 9
```

- the elements at indexes 0 and 2 have value 9,
- the elements at indexes 1 and 3 have value 3,
- the elements at indexes 4 and 6 have value 9,
- the element at index 5 has value 7 and is unpaired.

Your task is to write an algorithm to figure out which number left unpaired.

Assume that:

- **N** is an odd integer within the range [1..1,000,000];
- each element of array A is an integer within the range [1..1,000,000,000];
- all but one of the values in A occur an even number of times.

Input:

The first line of the input contains the number of test cases. Then several lines follow. Each case consists of N+1 numbers. The first number, N, inform about the size of the set. Then, N lines come with each element of the set.

Output

For each test case, your program will print the element that remains unpaired.

Examples

Input	Output
2 5 1 2 3 2 1 3 1 1 1	3 1

Problem D: Is bigger smarter?

Some people think that the bigger an elephant is, the smarter it is. To disprove this, you want to take the data on a collection of elephants and put as large a subset of this data as possible into a sequence so that the weights are increasing, but the IQ's are decreasing.

Input

The input will consist of data for a bunch of elephants, one elephant per line, terminated by the endof-file. The data for a particular elephant will consist of a pair of integers: the first representing its size in kilograms and the second representing its IQ in hundredths of IQ points. Both integers are between 1 and 10000. The data will contain information for at most 1000 elephants. Two elephants may have the same weight, the same IQ, or even the same weight and IQ.

Output

Say that the numbers on the i-th data line are `W[i]` and `S[i]`. Your program should output **the size, n**, of a sequence. Such a sequence if formed by **n** positive integer (each one representing an elephant). If these **n** integers are `a[1], a[2],..., a[n]` then it must be the case that

```
W[a[1]] < W[a[2]] < ... < W[a[n]]
```

and

```
S[a[1]] > S[a[2]] > ... > S[a[n]]
```

In order for the answer to be correct, **n** should be as large as possible. All inequalities are strict: weights must be strictly increasing, and IQs must be strictly decreasing. There may be many correct sequences for a given input, but given that the output if its length, the correct solution is unique.

Examples

Input	Output
6008 1300 6000 2100 500 2000 1000 4000 1100 3000 6000 2000 8000 1400 6000 1200 2000 1900	4

Explanation:

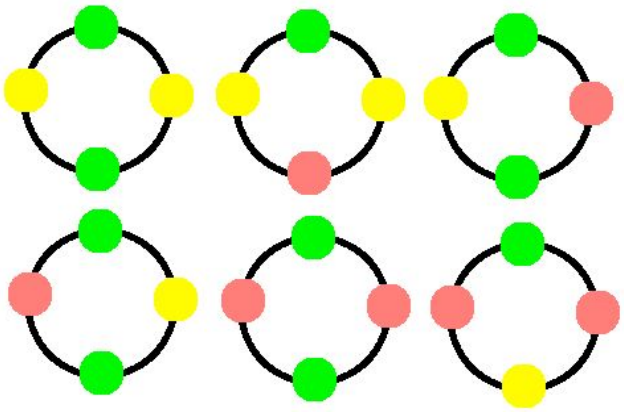
One largest sequence that satisfies the condition is: `4, 5, 9, 7`. And its size is 4, which is informed as the output.

Problem E: Painting

For her birthday Dasha got a beautiful string of beads. The beads were wonderful: **N** beads in total, each of them was colored in one of **K** different colors. They also had an interesting property: no two adjacent beads had the same color.
When Dasha put the beads on, a sudden thought struck here. What if there is another girl out there with exactly the same beautiful beads? To check how probable it is, Dasha wants to calculate the number of ways to paint **N** beads in not more than **K** colors, with no two adjacent beads having the same color.
Help Dasha out! Since the answer can be very large, return it modulo $10^9 + 7$.

Example

For N = 4 and K = 3, the output should be 18.
Let's assume that the beads are colored green, pink and yellow. If the green bead is the topmost one, there are 6 possible color combinations:



Since pink and yellow can also be on top, and for each of them there are also exactly 6 color combinations, the final answer is 6 * 3 = 18.

Input

The first line is an integer T, denoting the number of test cases. Then, T lines follow. Each test case consists of 2 integers: $2 \le \mathbf{N} \le 10^9$ and $1 \le \mathbf{K} \le 10^9$.

Output

For each test case, your program will print the number of possible combinations, modulo $10^9 + 7$.

Examples

Input	Output
2 4 3 2 1	18 0

Problem F: Vonny and her dominoes

Vonny loves playing with dominoes. And so she owns a standard set of dominoes. A standard set of dominoes consists of 28 pieces called bones, tiles or stones. Each bone is a rectangular tile with a line dividing its face into two square ends. Each square is labeled with a number between 0 and 6. The 28 stones are labeled (0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6), (1,1),(1,2),...,(5,5),(5,6),(6,6). Tommy - the brother of Vonny - build a box for Vonny's dominoes. This box is sized 7 x 8 squares. Every square is labeled with a number between 0 and 6. You can see a example box here.

```
0 3 0 2 2 0 2 3
1 5 6 5 5 1 2 2
3 4 1 4 5 4 4 4
6 6 1 0 5 2 3 0
4 0 3 2 4 1 6 0
1 4 1 5 6 6 3 0
1 2 6 5 5 6 3 3
```

Now Vonny wants to arrange her 28 stones in such way that her stones cover all squares of the box. A stone can only be placed on two adjacent squares if the numbers of the squares and of the domino stone are equal. Tommy asks Vonny in how many different ways she can arrange the dominos. Tommy assumes that Vonny need a lot of time to answer the question. And so he can take some of Vonny's candies while she solves the task. But Vonny is a smart and clever girl. She asks you to solve the task and keeps an eye on her candies.

Input

The first line of the input contains the number of testcases. Each case consists of 56 numbers (7 rows and 8 cols) between 0 and 6 which represents Tommy's box.

Output

For each testcase output a single line with the number which answers Tommy's question.

Example

Input	Output
2 0 3 0 2 2 0 2 3 1 5 6 5 5 1 2 2 3 4 1 4 5 4 4 4 6 6 1 0 5 2 3 0 4 0 3 2 4 1 6 0 1 4 1 5 6 6 3 0 1 2 6 5 5 6 3 3 5 3 1 0 0 1 6 3 0 2 0 4 1 2 5 2 1 5 3 5 6 4 6 4 0 5 0 2 0 4 6 2 4 5 3 6 0 6 1 1 2 3 5 3 4 4 5 3 2 1 1 6 6 2 4 3	18 1

Problem G: Peer Review

For scientific conferences, scientists submit papers presenting their ideas, and then review each other's papers to make sure only good papers are presented at the conference. Each paper must be reviewed by several scientists, and scientists must not review papers written by people they collaborate with (including themselves), or review the same paper more than once.

You have been asked to write a program to check if your favorite conference is doing things right. Whether a paper is being reviewed too much, too little, or by the wrong people - the organizers must know before it is too late!

Input

The first line in each test case has two integers, K (1 <= K <= 5) and N (1 <= N <= 1000). K is the number of reviews that each paper will receive, while N is the number of papers to be reviewed. The conference only accepts papers with a single author, and authors can only present a single paper at the conference.

Each of the next N lines describes an author and includes the name of the institution to which the author belongs, followed by the list of the K papers he or she has been requested to review. It is assumed that researchers from the same institution collaborate with each other, whereas researchers from different institutions don't. All institution names are shorter than 10 characters, and contain only upper or lowercase letters and no whitespace. Since we have as many papers as authors, papers are identified by their author's index; paper 1 was written by the first author in the list, and paper N was written by the last author.

The end of the test cases is marked with a line containing K = 0 and N = 0. You should generate no output for this line.

Output

For each test case, your program should output `NO PROBLEMS FOUND` (if all rules are being followed) or `P PROBLEMS FOUND`, where `P` is the number of rule violations found (counting at most 1 violation per paper). If there is exactly one rule violation overall, your program should output `1 PROBLEM FOUND`.

Example

Input	Output
2 3 UCM 2 3 UAM 1 3	NO PROBLEMS FOUND 3 PROBLEMS FOUND

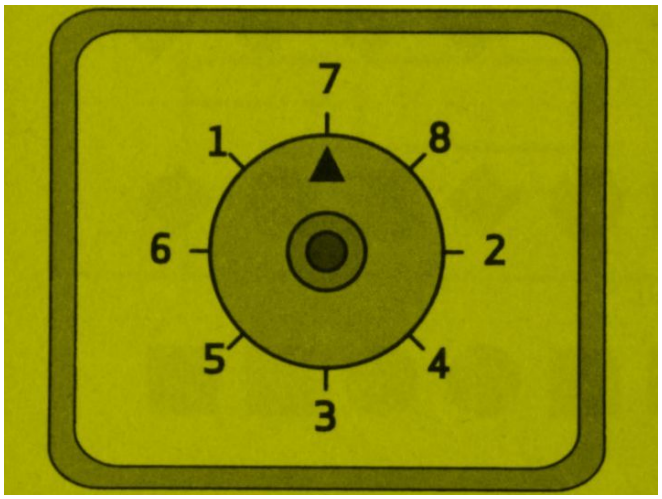
UPM 1 2 2 3 UCM 2 3 UAM 1 2 UPM 2 2 0 0	

Problem H: Combination Safe

A safe (also called a strongbox or coffer) is a secure lockable box used for securing valuable objects against theft and/or damage from fire. A safe is usually a hollow cuboid or cylinder, with one face being removable or hinged to form a door. The body and door may be cast from metal (such as steel) or formed out of plastic through blow molding. Bank teller safes typically are secured to the counter, have a slit opening for dropping valuables into the safe without opening it, and a time-delay combination lock to foil robbers/and or thieves. One significant distinction between types of safes is whether the safe is secured to a wall or structure or if it can be moved around. A less secure version (only suitable for petty cash) is usually called a cash-box.

Peter is an experienced locksmith but does not like working with numbers, so he asked you to help him to properly configure the locks on safes that in this time of insecurities, several companies are installing in his neighborhood.

He is working with a model of lock that have the following shape:



The combination of the safe is made up of six digits whose sum is X. The lock works as follows: starting with the upper digit (7); jump one digit to the left or to the right (1 or 8), from there jump two digits to the left or to the right, and then jump three, four and five digits on, also to left or to right.
Peter wants you code a program that receives as input the X value (20 <= X <= 40) and report the number combination that lets you open the box (should be unique). If there is more than one numeric combination your program will print "Duplicated" and if there isn't any numeric combination will print "Impossible".

Input

The input consits of an integer number X (20 <= X <= 40).

Output

Your program will show only one line containing the numeric combination (if it is unique) consisting of the sequence of six digits without spaces, “Duplicated” or “Impossible”, as appropriate.

Examples

Input	Output
40	Impossible
24	715731
25	Duplicated

Problem I : Peer Review

For scientic conferences, scientists submit papers presenting their ideas, and then review each other's papers to make sure only good papers are presented at the conference. Each paper must be reviewed by several scientists, and scientists must not review papers written by people they collaborate with (including themselves), or review the same paper more than once.
You have been asked to write a program to check if your favorite conference is doing things right. Whether a paper is being reviewed too much, too little, or by the wrong people - the organizers must know before it is too late!

Input

The first line in each test case has two integers, K (1 K 5) and N (1 N 1000). K is the number of reviews that each paper will receive, while N is the number of papers to be reviewed. The conference only accepts papers with a single author, and authors can only present a single paper at the conference.
Each of the next N lines describes an author and includes the name of the institution to which the author belongs, followed by the list of the K papers he or she has been requested to review. It is assumed that researchers from the same institution collaborate with each other, whereas researchers from different institutions don't. All institution names are shorter than 10 characters, and contain only upper or lowercase letters and no whitespace. Since we have as many papers as authors, papers are identied by their author's index; paper 1 was written by the rst author in the list, and paper N was written by the last author.
The end of the test cases is marked with a line containing K = 0 and N = 0. You should generate no output for this line.

Output

For each test case, your program should output NO PROBLEMS FOUND (if all rules are being followed) or PROBLEMS FOUND, where P is the number of rule violations found (counting at most 1 violation per paper).
If there is exactly one rule violation overall, your program should output 1 PROBLEM FOUND.

Examples

Input	Output
2 3 UCM 2 3 UAM 1 3 UPM 1 2 2 3 UCM 2 3 UAM 1 2 UPM 2 2 0 0	NO PROBLEMS FOUND 3 PROBLEMS FOUND

Problem J: Funny Plant

Scientist have discovered a new plant. The fruit of the plant can feed 1 person for a whole week and best of all, the plant never dies. Fruits needs 1 week to grow, so each week you can harvest it fruits. Also the plant gives 1 fruit more than the week before and to get more plants you need to plant a fruit.
Now you need to calculate after how many weeks, you can support a group of x people, given y fruits to start with.

Input

The first line is a single positive integer t. The next t lines contain 2 positive integers x and y, being x the number of people needed to be fed and y the number of fruits you start with, where 1 <= x, y <= 7,400,000,000 (the approximate world's population).

Output

For each pair of numbers x and y, indicate in a single line the number of weeks before you can feed the entire group of people.

Explanation:

Here you have a table that shows the growth when starting with 1 fruit. It shows when the plant came into existence (is planted) and how many fruit it bears each week.

```
  Plant 1  2  3  4  5  6  7  8  9 10 11 12 13    Total fruits for the week
Week
1       0  -  -  -  -  -  -  -  -  -  -  -  -     0
2       1  0  -  -  -  -  -  -  -  -  -  -  -     1
3       2  1  0  0  0  -  -  -  -  -  -  -  -     3
4       3  2  1  1  1  0  0  0  0  0  0  0  0     8
5       4  3  2  2  2  1  1  1  1  1  1  1  1    21
```

At week 1 we have 1 plant giving 0 fruits, because it has just been planted.
When week 2 comes along we have 1 plant that gives off a fruit and then we use that fruit to plant plant 2.
Then in week 3 we have 2 fruits from plant 1, 1 from plant 2, so we can plant 3 new plants.

Examples

Input	Output
3 200 15 50000 1 150000 250	5 14 9

Problem K: Switches

When you were a little kid, was indiscriminately flicking light switches super fun? I know it was for me. Let's tap into that and try to recall that feeling with today's challenge.
Imagine a row of N light switches, each attached to a light bulb. All the bulbs are off to start with. You are going to release your inner child so they can run back and forth along this row of light switches, flipping bunches of switches from on to off or vice versa. The challenge will be to figure out the state of the lights after this fun happens.

Input

The first line is a single positive integer t, which indicates the number of test cases that follows.
Each test case will have two parts. First, the number of switches/bulbs N (1 <= N <= 1000) and the number of passes of your inner child M (1 <= M <= 100) are specified in a single line. On the remaining lines, there will be pairs of integers indicating ranges of switches that your inner child toggles as he runs back and forth. These ranges are inclusive (both endpoints, along with everything between them, are included), and the positions of switches are zero-indexed (so the possible positions range from 0 to N-1).

Output

For each test case, the output will include a single line with a number: the number of switches that are on after all the running around.

Examples

Input	Output
1 10 4 3 6 0 4 7 3 9 9	7

There is a more thorough explanation of what happens below.

Explanation of example

Below is a step by step rendition of which switches each range toggled in order to get the output described above.

```
    0123456789
    ..........
3-6    ||||
    ...XXXX...
0-4 |||||
    XXX..XX...
7-3    |||||
    XXXXX..X..
9-9          |
    XXXXX..X.X
```

As you can see, 7 of the 10 bulbs are on at the end.