The 2017 ACM-ICPC

PolyU International Invitaiton Contest

August 25th, 2017

Problem Set

Problem	Time Limit (sec)
A. Anniversary	1.000
B. Currency	0.250
C. Slogan	1.000
D. Go	1.000
E. Game	3.000
F. Bitmap	12.000
G. Ivan Comes Again	2.000
H. Jerry Mouse	1.000
I. Invertible Tree	2.000
J. Fraction Calculator	1.000

Contest Details and Rules

1 Judging Environment

1.1 OS

- Ubuntu 16.04.1 LTS 64-bit

1.2 Languages

1.2.1 C

1. Compiler:

   gcc (Ubuntu 5.4.0-6ubuntu1~16.04.4) 5.4.0 20160609

2. Compilation:

   gcc -g -O2 -std=gnu99 -static {Submission} -lm

3. Submissions:

   *.c

1.2.2 C++

1. Compiler:

g++ (Ubuntu 5.4.0-6ubuntu1~16.04.2) 5.4.0 20160609

2. Compilation:

   g++ -g -O2 -std=gnu++14 -static {Submission}

3. Submissions:

   *.cc / *.cpp / *.cxx / *.c++ / *.C

1.2.3 Java

1. Compiler:
   1. openjdk version "1.8.0_131"
   2. OpenJDK Runtime Environment (build 1.8.0_131-8u131-b11-2ubuntu1.16.04.3-b11)
   3. OpenJDK 64-Bir Server VM (build 25.131-b11, mixed mode)
2. Compliation:
   1. javac -encoding UTF-8 -sourcepath . -d . {Submission}
   2. java -XX:UseSerialGC -Xss64m -Xms1024m -Xmx1024m
3. Submissions:

   *.java

1.3 Control System

- PC^2 9.4.1

2 Coding Environment

- The coding environment of this contest will be almost identical to that of 2017 ACM-ICPC World Finals. The contest image of 2017 ACM-ICPC World Finals could be found here. The only difference is this contest will use PC^2 as the online judge system instead of KATTIS.

2.1 OS & Languages & Control System

- The same as Judging Environment.

2.2 Desktop

- GNOME

2.3 Editors

- VIM / GVIM
- Emacs
- Gedit
- Geany

2.4 IDEs

2.4.1 C / C++

- Eclipse 4.6 + CDT 9.0.1

2.4.2 Java

- Eclipse 4.6

A. Anniversary

2017 is the 80th year anniversary of PolyU, some students will celebrate for the anniversary. So, they decided to make a huge billboard which will be hanged on the tall buildings in Yuk Choi Street around the campus. Now assume the building in the street has different height and the width are all 1, and the building are next to others closely. The shape of the billboard is rectangle. Students want to make the huge billboard as large as possible. Please help them and give the correct answer of maximum **AREA** of the billboard.

Input

There are multiple test cases in this problem.

In the first line, a number T (T<=100) will be given which means there are T test cases.

In each test case, there will be two lines. In the first line is a single number N(N<=1,000), which means there are N building in the street and in the second one there will be N numbers which means the height of each building.

We guarantee there will be at most 10 test cases that N is larger than or equal to 500 and all buildings' heights will never exceed 10,000.

Output

The output contains T lines.

In each line, output the maximum square of the billboard of each test cases.

Samples

Anniversary.in	Anniversary.out
2 5 5 5 5 5 7 8 2 3 4 5 5 4 3 2	25 16

Hint

In the second test cases, we can make the width and height of the billboard 6 and 3 respectively, and that can be hanged between the 2nd and the 7th building and the square is 18. It can be proved the square of billboard is the maximum in this problem.

B. Currency

When you come to Hong Kong, the first thing is currency exchange because of foreign currencies are not accepted in Hong Kong. Now, you are given some kinds of currencies, the exchange rate and the amount of Hong Kong Dollars you need. Please calculate the amount of foreign currencies need.

Input

The input contains N+2 lines.

In each test cases, the first line contains a number N (N <= 15), which means the number of foreign currencies, and N lines follows. In the following N lines, each line contains a string and a real number, which indicate the name of the foreign currency and the exchange rate which expressed in 100 units of the foreign currency against HKD.

And the last line contains a real number which means the amount of HKD in needed.

Output

The output contains N lines.

Each line contains the name of foreign currency and the amount needed. Real numbers in the outputs should be rounded to 0.01.

The output order of currencies should be the same as the input one.

Samples

Currency.in	Currency.in
8 CNY 112.42 USD 776.22 GBP 973.50 AUD 589.95 CAD 587.70 EUR 826.20 CHF 774.05 DKK 111.10 1500.00	CNY 1334.28 USD 193.24 GBP 154.08 AUD 254.26 CAD 255.23 EUR 181.55 CHF 193.79 DKK 1350.14

C. Slogan

There is a slogan on Panama Canal, “A man a plan a canal Panama.” We can find after excluding the dots and spaces, the letters are same even if read from left to right or from tight to left. Slogans which contains the property is called “Panama slogan”. Now you are given multiple slogans, please check whether it is a “Panama slogan”.

Input

The input contains N+1 lines.

Line 1: a number N, which means the number of slogans.

Line 2-N+1: each line contains a slogan, we guarantee the slogan only contains Latin letters, spaces, commas and dots.

Output

The output contains N lines.

For each slogan, if it is a “Panama slogan”, output “YES”, otherwise, output “NO” instead.

Samples

Slogan.in	Slogan.out
3 The Hong Kong Polytechnic	NO YES

University. Go Og A man a plan a canal Panama.	YES

Data Range:

We guarantee N does not exceed 1,000 and the length of each slogan does not exceed 1,000.

D. Go

On the end of May 2017, the Go competition between Ke Jie and AlphaGo was held on Wuzhen, China. Finally, AlphaGo beat Ke Jie with a score 3:0. Now you are asked to design a machine to check who is the winner.

Now a simple rule of Go is given by Wikipedia:

Rule 1. Go is a game between two players, called Black and White.

Rule 2. Go is played on a plane grid of 19 horizontal and 19 vertical lines, called a *board*.

Rule 3. Go is played with playing tokens known as *stones*. Each player has at their disposal an adequate supply of stones of their colour.

Rule 4. At any time in the game, each intersection on the board is in one and only one of the following three states: 1) empty; 2) occupied by a black stone; or 3) occupied by a white stone. A *position* consists of an indication of the state of each intersection.

Rule 5. If one player has a higher score than the other, then that player wins. Otherwise, the game is drawn.

Rule 6. Because Black goes first, so the score of Black need to be deducted by 7.5 points.

After a match finished the intersection can be one and only one of the following three two states: 1) occupied by a black stone or 2) occupied by a white stone.

Now you are given the board state of a match, please check who is the winner or the match has not finished yet.

Input

The input has multiple cases.

Each case has only one line, containing two numbers x and y, which indicated the number of intersections occupied by black and white respectively.

Output

For each input cases, output one line.

If black wins, output "Black";

If white wins, output "White";

If the match has not finished yet, output "Unknown".

Go.in	Go.out
190 171 200 150 0 361	Black Unknown White

E. Game

Most people have heard of Tic-Tac-Toe. Two players use two different marks to take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.

Now, Alice and Bob found the game is boring because if you know the strategy, the result will always go to a tie. So, they decided to play it on 4*4 grid where other rules remain the same. Now it is Alice's turn, please check if Alice is sure to win. Please notice that, Alice always use 'x' and goes first while Bob always use 'o' and goes second.

Input

The first line contains a number T, and T test cases follows.

Each case contains 4 lines and each line contains 4 characters. The character can only be one of 'x', 'o', and '.'

'x' means the point was taken by Alice.

'o' means the point was taken by Bob.

'.' Means the point was blank now.

Please notice that we number the top-left point to be (0, 0) and the bottom-right to be (3, 3). i.e. The coordinator (3, 0) means the

bottom-left point.

Output

For each input case, firstly output “Case #k:” where k is the kth input (numbered from 1). Then if there are any point Alice can be sure to win, output all of them. One point per line and ordered from top to bottom, if two or more points are on the same line, order them from left to right. Otherwise, output “#####” instead (without quotes).

Samples

Game.in	Game.out
2 xxx. xxx. .ooo ooo. …. .xo. .ox. ….	Case #1: (0, 3) (1, 3) Case #2: #####

F. Bitmap

Image Recognition Technology has become a fast-growing technology in the recent few years. Now you are asked to make a 3D image translator to translate 3D image to a computer-readable string.

Rule 1: The 3D image only has two colours, black and white.

Rule 2: The 3D image has the same length in its length, width and height, all of them are 2^N pixels.

Rule 3: If the colour in all pixels in an image is black or white, it will be translated to “1” or “0” respectively, otherwise, it will be separate to 8 sub-image which have the same size, and the translation result is the character ‘*’ following with 8 sub-images’ translation result. The order of the 8 sub-images result following the order: (1) The image which has the larger number in z axis will output first. (2) If two or more sub-images has the same number in z-axis, then following the Quadrant order.

E.g.

An image with 2 pixels in its length, width and height, the output order of sub-image is:

(2,2,2) (1,2,2) (1,1,2) (2,1,2) (2,2,1) (1,2,1) (1,1,1) (2,1,1)

Input

The input has multiple cases.

The first line is a number T(T<=100), which means there are T test cases in total.

In each test cases,

The first line contains a number N(N<=7), which means the length, width and height are all 2^N pixels.

The second line contains a number M(M<=100,000), which means there are M pixels are black in totals.

The following M lines, each line contains three number x,y,z, which means that the pixels (x, y, z) is blacks.

The input file does not exceed 30 Megabytes.

Ramark:

The coordinator (x, y, z) may appear more than once.

Output

For each test cases, output the translation result.

Bitmap.in	Bitmap.out
2 1 8 1 2 2	*11111100 *10000010

1 1 2 1 2 1 2 2 2 2 1 2 2 1 2 2 2 1 1 2 2 1 2 1 1 1 2 2 2	

G. Ivan comes again!

There is a large matrix whose row and column are less than or equal to 1000000000. And there are three operations for the matrix:

1) *add*: Mark an element in the matrix. The element wasn't marked before it is marked.

2) *remove*: Delete an element's mark. The element was marked before the element's mark is deleted.

3) *find*: Show an element's row and column, and return a marked element's row and column, where the marked element's row and column are larger than the showed element's row and column respectively. If there are multiple solutions, return the element whose row is the smallest; and if there are still multiple solutions, return the element whose column is the smallest. If there is no solution, return -1.

Of course, Saya comes to you for help again.

Input

The input consists of several test cases.

The first line of input in each test case contains one integer N ($0<N\leq 200000$), which represents the number of operations.

Each of the next N lines containing an operation, as described above.

The last case is followed by a line containing one zero.

Output

For each case, print the case number (1, 2 …) first. Then, for each "*find*" operation, output the result. Your output format should imitate the sample output. Print a blank line after each test case.

Sample

4 add 2 3 find 1 2 remove 2 3 find 1 2 0	Case 1: 2 3 -1

H. Jerry Mouse

Kudo and Saya are good friends, and they are always together.

But today, since Saya is not here (Where is Saya? You can get the answer at Problem F), Kudo fells very bored. So Kudo starts to watch TV for fun.

Now, she is watching the famous cartoon “Tom and Jerry”. Jerry’s home have many entrance, he always enter his home in an entrance and get out from another.

Kudo suddenly thought that what will happen if there are many Jerrys?

Kudo finds out a paper box and dig many holes in the side of the box. Then, she marks every hole with an integer, representing its owner. Finally, she signs the box “Kudo’s House” as shown in Figure 1.

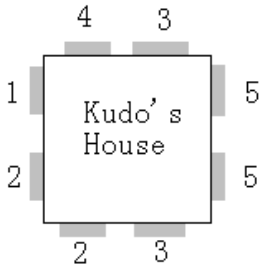

Figure 1 *The gray part represents a hole, and the numbers means its owner, i.e. the two gray parts in the lower left corner means the two holes belongs to mouse 2.*

Kudo think it is very interesting, so she makes a lot of boxes, and sign them “Saya’s House”, “Riki’s House”, “Lin’s House” and so on.

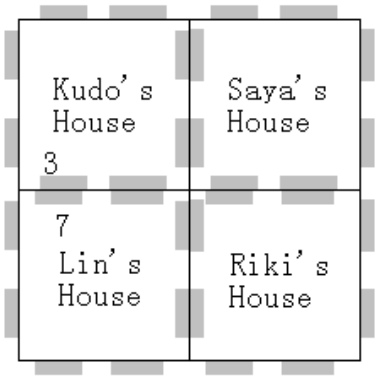

Figure 2 *The jointed gray parts represent the same hole.*

At last, she combines them together to make a large box as shown in Figure 2.

She defined mouse A and mouse B is the same mouse if and only if:

a) mouse A and mouse B are in a same house and A equals to B;
b) mouse A and mouse B are in different houses, but they have a hole that combined together, such as mouse 3 in Kudo’s House and mouse 7 in Lin’s House;
c) There exits a lists of mouse M1, M2 … Mp, while A and M1 are the same, M1 and M2 are the same … Mp and B are the same.

But there is a problem: Suppose mouse 3 in Kudo’s House is the same with mouse 7 in Lin’s House, while mouse 7 in Lin’s House is the same with mouse 4 in Kudo’s House. It means that mouse 3 and 4 in Kudo’s House are the same!

Kudo is very confused, so she comes to you for help.

Given the initial *N* boxes, can you tell her finally each hole belongs to whom?

Here, *N* always equals to 1, 4, 9, 16, 25, 36, 49, 64, 81 or 100. Kudo always combines the boxes as shown in Figure 3, while $n = \sqrt{N}$.

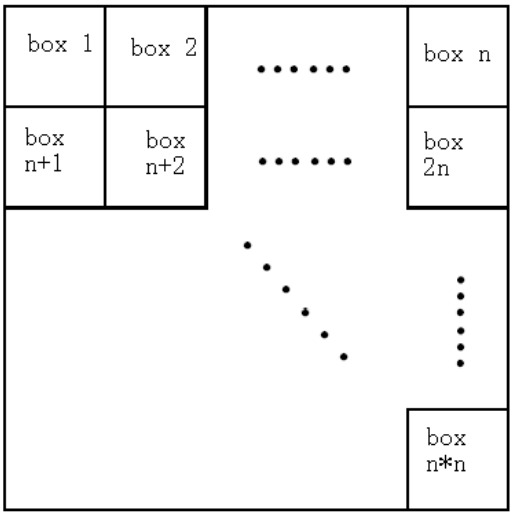

Figure 3 *How Kudo combines the boxes.*

Input

The input consists of several test cases.

The first line of input in each test case contains two integers *N* (0<*N*≤100) and *M* (0<*M*≤1000), which represent the number of boxes and the number of holes in each side of the box. You can assume that there are always *M* holes in every side of the *N* boxes.

Each of the next *N* lines containing *4M* integers representing the holes on the boxes. The first *M* integers represent the holes on the upside, from left to right; the second *M* integers represent the holes on the

downside, from left to right; the third *M* integers represent the holes on the leftward, from up to down; the forth *M* integers represent the holes on the rightward, from up to down. You can assume that the hole number is not greater than *M**2.

The last case is followed by a line containing two zeros.

Output

For each case, print the case number (1, 2 …) and *4*n*M* integers ($n=\sqrt{N}$) represents the holes on the upside, downside, leftward and rightward side of the large box, using the same format as the input file.

In your answer, the holes signed by the same numbers should belong to the same mouse, while the holes signed by different numbers should belong to different mouse. The number should be an integer, starting from 1. Since there are multiply solutions, please print the one whose first number is the least. If there are still multiply solutions, print the one whose second number is the least, and so on.

Your output format should imitate the sample output. Print a blank line after each test case.

Sample

4 1 2 2 1 1	Case 1: 1 2 1 2 3 4 3 4

2 2 1 1 1 1 2 2 1 1 2 2 0 0	

I. Invertible Tree

Write a program to perform two types of operations on the given rooted tree where nodes can be either red or black. One types of operations is "inversion" which intends to change a node and all of its descendants to their other colour respectively. The other types of operations is "query" which means to ask the present colour of a node. In the beginning, the colour of all the nodes is red.

Input

The number of nodes 'n' and the total number of operations 'm', where n and m <= 100000.

Then n-1 lines of edges. Each of them consists of the first parent node and second the child node.

Finally m lines of operations. Every operation begins with a character which is 'I' or 'Q'. 'I' represents "inversion" and 'Q' stands for "query". They are followed by their respective node.

Output

The answers of the "query" operations. Begin a new line when answering a new "query".

Sample

11 13	R
1 2	B
1 3	R
3 4	B
3 5	R
5 6	B
5 7	R
7 8	
7 9	
9 10	
9 11	
Q 11	
I 1	
Q 11	
I 3	
Q 11	
I 5	
Q 11	
I 7	
Q 11	
I 9	

Q 11 I 11 Q 11	

J. Fraction Calculator

Design a calculator to solve an expression consisting of fractions and four operators, '+', '-', '*' and '/'! No '(' or ')' will present in the expression. Fractions will be given in the form of "a/b" where 'a' is the numerator and 'b' is the denominator. A '-' needs to be added in front of the notation when the fraction is negative. In the case of 0, the fraction needs to be writen as "0/1".

Input

A line of an expession. It is guaranteed that no divisor will equal 0 in the whole process of calculation. We guarantee that the length of the expression does not exceed 100.

Output

The answer of the expression in its lowest terms, which means that there is no common factor except 1 from its numerator and denominator.

Sample input

6/2 - 1/8 / 2/2 - -8/2 / 1/6	215/8