2015 Consortium for Computing Sciences in Colleges
Programming Contest
Saturday, November 7th
Roanoke College
Roanoke, VA

There are eight (8) problems in this packet. Each team member should have a copy of the problems. These problems are NOT necessarily sorted by difficulty. You may solve them in any order.

Remember input/output for the contest will be from `stdin` to `stdout`. `stderr` will be ignored. Do not refer to or use external files in your source code. Extra white space at the end of lines is ignored, but extra white space at the beginning or within text on a line is not ignored. An extra blank line of output is ignored, but blank lines at the beginning or between lines of text are not ignored.

Have A Lot Of Fun & Good Luck! ☺

Problem 1

Letters Not Used

For this problem, you are interested in displaying a list of all letters that do not appear in a given input string. The letters in the output will be lowercase, regardless of the case of the letters in the input string.

Input

The first line of input contains a single integer n, $(1 \leq n \leq 1000)$, which is the number of test cases that follow. Each test case consists of a single line of input containing a string. Each string is of length n, $(1 \leq n \leq 80)$. All input strings will consist solely of letters and spaces, no punctuation or special symbols.

Output

For each case, you should print out the statement :

`Letters missing in case x:`

where `x` is the case number. Place a colon immediately after the case number along with a single space. This is followed by all the letters which are not used in the input. All letters should be lowercase and in ascending sorted order from a to z.

Sample Input

```
5
The quick brown fox jumped over the lazy dog
How much better wood would a woodchuck chuck today than yesterday
A skunk sat on a stump and thunk the stump stunk
Pack my box with five dozen liquor jugs
Razorback jumping frogs can level six piqued gymnasts
```

Output Corresponding to Sample Input

```
Letters missing in case #1: s
Letters missing in case #2: fgijpqvxz
Letters missing in case #3: bcfgijlqrvwxyz
Letters missing in case #4:
Letters missing in case #5: hw
```

Problem 2

Welding Art

Josh is both an amazing artist and welder. He loves taking simple shapes and combining them to create simple, yet beautiful art. For instance, he will sometimes take circular arcs and weld them end-to-end on a flat surface giving a unique pattern.

Josh, however, wonders about the limits of this art form. For instance, if he takes one-third of the circle segments and welds them together, how many ways can they connect so that the last piece connects to the first? If Josh has three segments, the answer is obvious: two. Josh can weld the pieces by turning to the left or right, and they will meet back up at the end. However, for more pieces, the answer is not as obvious.

So, Josh has requested that you figure out how many different patterns he can create if he is using one-kth *(1/k)* of a circle segment and has n pieces, ALL of which he must use. Basically, each segment is an arc of a circle. So if $k = 5$, then the segment is a fifth of a circle segment.

Input

The input will start with a single positive number $c \leq 15$, that specifies the number of cases to follow. The next c lines will have two integers n ($1 \leq n \leq 30$), and k ($3 \leq k \leq 7$).

Output

For your output, you should simply print the number of patterns that Josh can create using n pieces of one-kth *(1/k)* circle metal.

Sample Input

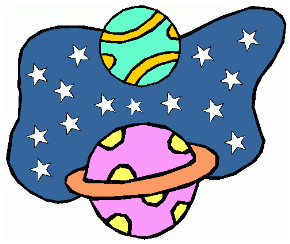
```
4
3 3
6 3
4 4
8 4
```

Output Corresponding to Sample Input

```
2
8
2
18
```

Problem 3

Planet Mining



Given the successfulness of comet landings and planetary fly-bys, NASA is attempting to identify the feasibility of mining the atmospheres of various planetoids for valuable minerals. Before launching a spacecraft, they'll need to simulate the mining process, and that's where you'll come in!

The atmospheres of most planets are made of a variety of materials, and so any desired material would not make up 100% of the entire atmosphere. The mining prototype, however, can account for this and extract the desired material out of the atmosphere, while also returning the remaining gas back into the atmosphere AT THE SAME VOLUME (by replacing it with an inert replacement gas). Because of this, the volume of the entire atmosphere does not change, but the relative concentration of the material being mined slowly depletes.

It is not cost effective to continue extracting materials once the concentration gets too low, so NASA wants you to calculate how long a mining craft should operate before it leaves the planet.

Input

The first line of input contains a single integer n, ($1 \leq n \leq 1000$), which is the number of test cases that follow. The next n line(s) will have four numbers: the starting concentration ($0.25 \leq k\_0 \leq 1.0$), the ending concentration ($0.001 \leq k\_end \leq 0.01$), the volume of the atmosphere in km^3 ($1 \leq v \leq 10$), and the rate at which the probe can purify the atmosphere in km^3/h ($1 \leq f \leq 10$).

Output

For your output, you should print out rounded to three decimal places how many hours it will take to extract materials to the point that the concentration has reached the ending concentration.

Sample Input

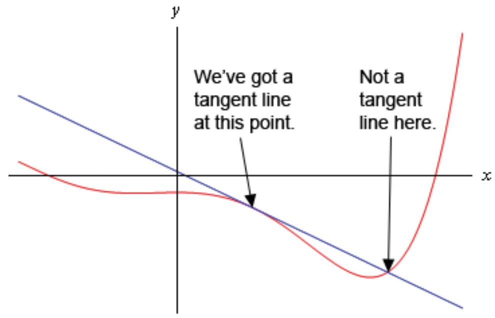
```
1
0.25 0.01 1 1
```

Output Corresponding to Sample Input

```
3.219h
```

Problem 4

Tangent Lines



Mary loves calculus, but there is one type of problem in particular that she would like to practice repeatedly to prepare for her next test. You will write a program to help her.

Consider the graph of $y = x^2$, which happens to be a parabola. Let the ordered pair (a, b) represent a point that lies "below" this parabola. In other words, assume that $a^2 > b$. Then, there exist two lines that pass through (a, b) that are each tangent to the graph of $y = x^2$. Your job is to compute the slopes of these two lines.

An instance of this problem is an ordered pair (a, b), where a and b are real numbers (numbers that can be found on a number line) such that $-100 \le a \le 100$ and $-100 \le b \le 100$. You may assume that $a^2 > b$. For each instance, you are to calculate the slopes of the two tangent lines that pass through (a, b) and which are tangent to the graph of $y = x^2$.

Input

The first line of the input will be the number of test cases. Each subsequent line contains a single ordered pair, including parentheses and a comma.

Output

For each test case your program should print a line of this form:

`Case <test case number>:  The slopes are <slope1> and <slope2>`

The `<test case number>` is a positive integer starting with 1. The real numbers `<slope1>` and `<slope2>` should be printed to exactly 2 decimal places. The value of `<slope1>` should be greater than `<slope2>`. Be sure to print 2 spaces after the colon, and do not put a period at the end of the sentence.

Sample Input

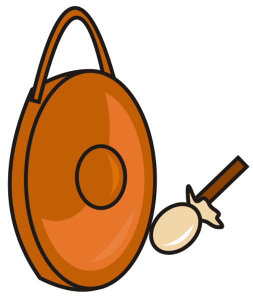
```
3
(0, -9)
(-6, 1)
(8, 2)
```

Output Corresponding to Sample Input

```
Case 1:  The slopes are 6.00 and -6.00
Case 2:  The slopes are -0.17 and -23.83
Case 3:  The slopes are 31.75 and 0.25
```

Problem 5

The Village Gong



In a certain remote village, monks announce the passage of time by striking a gong every ten minutes throughout the day and night. The gong is struck at exactly midnight, and at every ten minute interval after that, at 12:10:00, 12:20:00, etc. Your job is to figure out how long it will be until the monks will next strike the gong.

Input

The first line of the input will be the number of test cases. On each subsequent line is a separate test case. An instance of this problem is a string of the form `<h>:<m>:<s>` that represents the current time in a 12-hour format, where h, m and s are integers and $1 \leq h \leq 12$, $00 \leq m \leq 59$, $00 \leq s \leq 59$. The abbreviations "a.m." and "p.m." will not be specified in the input. They are unnecessary.

Output

For each instance, your program is to determine the integer number of seconds until the next strike of the gong. If the current time happens to be the precise moment when the gong is being struck, then output 0.

Sample Input

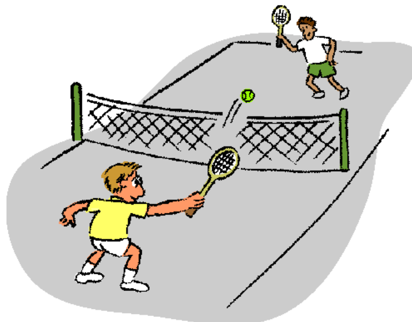
```
4
8:11:47
12:05:00
7:19:59
4:50:00
```

Output Corresponding to Sample Input

```
493
300
1
0
```

Problem 6

Tournament Seeds



It's time for a big tennis tournament! Three tennis players from your college are competing in a national tournament. Before the first round starts, we are curious to find out who the opponents will be for each of your three players. This program will calculate tournament seeds for the first round.

Let n be the number of players in a tennis tournament. Let us assume that $8 \le n \le 256$. Also, assume that every player in the tournament is assigned a seed, which is a unique number between 1 and n, inclusive.

If n is exactly a power of 2, then it is easy to calculate the seed of a given player's opponent. For any two players who play against each other in their first round, the sum of their seeds is invariant. The value of this invariant is equal to the sum of the highest and lowest seeds. Since n is a power of 2, the highest seeded player is number 1, and the lowest seeded player is number n. Thus, the invariant equals $n + 1$. So, if your seed is s, then the seed of your opponent is $(n + 1) - s$. For example, in a tournament of 16 players, the opponent of the #2 seed is $(16 + 1) - 2 = 15$, in other words the #15 seed.

However, what happens if n is not a power of 2? In this case, the top players get "byes" which means they do not play in the first round. They automatically qualify for the second round of the tournament. The players who do not get byes must play in the first round. The number of players who must play in the first round is 2^k, where k is an integer, and 2^k is the largest power of 2 less than n. Thus, the top $(n - 2^k)$ players get byes in the first round. For the other 2^k players, we can calculate seeds in a straightforward way. Again, for any two players in a matchup, the sum of their seeds is an invariant. And once again, this invariant is the sum of the highest and lowest seeds of the players who must play in the first round.

For example, if the tournament has 10 players, then the top 2 players receive byes. Seeds 3 through 10 must play in the first round. Thus, the invariant is $3 + 10 = 13$. Who is the opponent of the 3rd seed? Since the sum of the seeds is 13, the opponent's seed is $13 - 3 = 10$, which means the #10 seed.

Input

An instance of this problem refers to a single tennis tournament and consists of 4 numbers. The first number is the number of players in the tournament. The remaining 3 numbers are the seeds of players from a particular college. Your program will calculate the seeds of the opponents of each of these 3 players, and indicate if in any case there is a bye instead of an opponent.

The first line of input is the number of test cases (i.e. the number of tennis tournaments). You may assume there will be no more than 100 tournaments. Each subsequent line of input contains the 4 numbers representing a test case.

Output

In your output, you need to identify each tournament by number, starting with Tournament 1. Information about the seeds of a tournament must be indented 2 spaces, and must be in one of two possible forms, as appropriate:

```
  Seed <my seed number> gets a bye.
```

or

```
  Seed <my seed number> plays seed <opponent's seed number>.
```

Be sure to conclude each sentence with a period.

Sample Input

```
2
48 11 30 48
10 3 4 5
```

Output Corresponding to Sample Input

```
Tournament 1
  Seed 11 gets a bye.
  Seed 30 plays seed 35.
  Seed 48 plays seed 17.
Tournament 2
  Seed 3 plays seed 10.
  Seed 4 plays seed 9.
  Seed 5 plays seed 8.
```

Problem 7

Word Scramble

This problem involves strings made up of uppercase letters. Your job is take a single string and return the scrambled version of that string according to the following rules.

- The scrambling process begins at the first letter of the word and continues from left to right.
- If two consecutive letters consist of an "A" followed by a letter that is not an "A", then the two letters are swapped in the resulting string.
- Once the letters in two adjacent positions have been swapped, neither of those two positions can be involved in a future swap.

Input

The first line of input contains a single integer n, ($1 \leq n \leq 1000$), which is the number of test cases that follow. Each test case consists of a single line of input containing a string. Each string is of length n, ($1 \leq n \leq 80$) and is composed solely of uppercase letters.

Output

For each case, you should print out the scrambled word on a single line.

Sample Input

```
8
TAN
ABRACADABRA
WHOA
AARDVARK
EGGS
A
ABBAAC
BABACA
```

Output Corresponding to Sample Input

```
TNA
BARCADABARA
WHOA
ARADVRAK
EGGS
A
BABACA
BBACAA
```

Problem 8

Diverse Matrix?

A two-dimensional array is *diverse* if no two of its rows have entries that sum to the same value. In the following examples, the first array is diverse because each row sum is different, but the second array is not diverse because the first and last rows have the same sum.

1	3	2	7	3
10	10	4	6	2
5	3	5	9	6
7	6	4	2	1

1	1	5	3	4
12	7	6	1	9
8	11	10	2	5
3	2	3	0	6

Input

The first line of input contains a single integer n, ($1 \leq n \leq 1000$), which is the number of test matrices that follow. For each matrix being tested, the input will be a single line containing two integers separated by a space representing the number of rows r and columns c in the range between 1 and 10 inclusive, ($1 \leq r, c \leq 10$). This is followed by r rows each of c integers each separated by a space. All input is in row major order. Each integer is a valid 32-bit integer.

Output

For each matrix, output a single word `yes` or `no` in lowercase indicating whether it is diverse.

Sample Input

```
2
4 5
1 3 2 7 3
10 10 4 6 2
5 3 5 9 6
7 6 4 2 1
4 5
1 1 5 3 4
12 7 6 1 9
8 11 10 2 5
3 2 3 0 6
```

Output Corresponding to Sample Input

```
yes
no
```