A: Binary is the Best!

We all know that computers think in binary--if only people could also! In fact, in Star Trek Next Generation, the *Binars* were an alien race that once took over the Enterprise by talking to each other in binary. On Earth, the binary system of the I Ching, a text for divination, is based on the duality of yin and yang, reportedly invented by ancient King and philosopher named Fuxi. Of course, other number bases are important too. Haven't we all counted in base 5 recently? So, we need a program that converts binary to other number bases (between 3 and 10). We will assume that all the binary numbers are unsigned.

Input

The first line of input is an integer (1 <= T <= 30) indicating the number of test cases to follow. Each test case consists of an integer base (between 3 and 10), followed by a binary number between 1 and 31 digits long.

Output

For each input test case, your program must output the converted number, printed in the specified base numbering system.

Sample Input

```
4
10 101011
8  101011
4  101011
7  1110001
```

Sample Output

```
43
53
223
221
```

E: Composition

Functions are everywhere! We use them as shorthand for complicated expressions or algorithms, and we use them to provide readable divisions for our code. However, in this problem, we'll be looking at the functions you know and love from math. You will be given a set consisting of input and output pairs for the function, which will be a one-toone function from the domain set **{1, 2, …, n}** to the range set**{1, 2, …, n}**. Being one-to-one means that every element in the domain is mapped to one and only one element in the range, and every element in the range is mapped to one and only one element in the domain. Your task is to find the smallest m such that for any element **x** in the domain set

We have that $\mathbf{f^m(x) = x}$. where $\mathbf{f^m(x)}$ is the function composed m times. For example, if **m = 3**, we have $f^3(x)$ = f(f(f(x))).

Input

Input begins with the number of test cases T (T ≤ 100). Each test case will begin with the number of elements in the set N (N ≤ 100,000) and will be followed by N lines where each line will have two numbers A and B (1 ≤ A, B ≤ 100,000) which will be the input/output pair (that is, f(A) = B)

Output

For each test case, print out the value of m. Your output should follow the exact format shown in the sample output below.

Sample Input

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

Sample Output

```
2
6
```

Problem E: Letter Cubes

Source file: `cubes.{c, cpp, java}`
Input file: `cubes.in`

This problem is based on a puzzle by Randall L. Whipkey.

In the game of Letter Cubes, there are a set of cubes, with each face of each cube having a letter of the alphabet, such that no letter appears more than once within the entire set. The maximum number of cubes is 4, allowing for up to 24 of the 26 letters of the alphabet to occur.

Words are formed by rearranging and turning the cubes so that the top letters of all the cubes together spell a word. The 13 words below have been made using a particular set of cubes.

CLIP
CLOG
CONE
DISH
FAZE
FURL
MARE
MOCK
QUIP
STEW
TONY
VICE
WARD

Only 23 distinct letters were used in the above words, so we will tell you the extra information that a B is included on one cube. Can you now determine the letters on each cube? For the above set of words, there is indeed a unique set of cubes. We will state this solution in canonical form as

ABCHTU DEKLQY FGIMNW OPRSVZ

Note that the letters on each individual cube are stated as a string of characters in alphabetical order, and the four 6-letter strings representing the four cubes are also listed in alphabetical order.

A simpler example relies on two cubes, forming the following 11 two-character strings (although the puzzles are more fun when the strings are actual words, they do not need to be):

PI
MU
HO
WE
WO
BE
MA
HI
RE
AB
PY

The only solution for the two cubes forming these strings is

AEIOUY BHMPRW

The same two cubes could be determined without the last pair PY being listed, as long as you were told that there was a Y on one cube. Your job is to make similar deductions.

Input: The input will contain from 1 to 20 datasets. The first line of each dataset will include a positive integer n ($6 \le n \le 30$) and a character c, described below. The next n lines will each contain a string of uppercase letters. Each string will be the same length, call it k, with $2 \le k \le 4$. Following the last dataset is a line containing only 0.

Returning to the issue of the special character, c, on the first input line for each dataset, there will be two cases to consider. Recall that the implicit set of k cubes must use $6*k$ distinct letters on their collective faces If all $6*k$ of those letters appear within the set of strings, then the character c on the first line of input is a hyphen, '-'. Otherwise, the strings have been chosen so that only one letter on the cubes does not appear. In this case, the character c on the first line of input will be that undisplayed letter. (For example, the `B` in our opening puzzle.)

Output: There is one line of output for each dataset, containing a 6-letter string for each cube, showing the letters on the faces of that cube. Each of those strings should have its letters in alphabetical order, and the set of strings should be given in alphabetical order with respect to each other, with one space between each pair. *We have chosen datasets so that each has a unique solution.*

Example input:	**Example output:**
13 B CLIP CLOG CONE DISH FAZE FURL MARE MOCK QUIP STEW TONY VICE WARD 11 - PI MU HO WE WO BE MA HI RE AB PY 10 Y PI MU HO WE WO BE MA HI RE AB 0	ABCHTU DEKLQY FGIMNW OPRSVZ AEIOUY BHMPRW AEIOUY BHMPRW

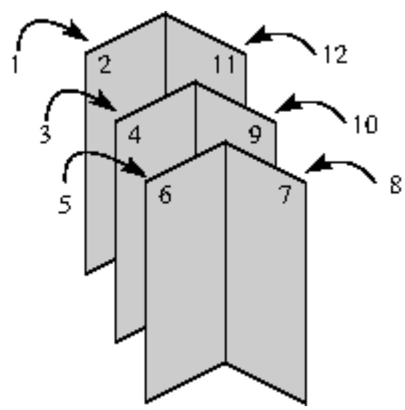
ACM Mid-Central Programming Competition 2013

Problem C: Missing Pages

Source file: `missing.{c, cpp, java}`
Input file: `missing.in`

Long ago, there were periodicals called *newspapers*, and these newspapers were printed on *paper*, and people used to *read* them, and perhaps even share them. One unfortunate thing about this form of media is that every so often, someone would like an article so much, they would take it with them, leaving the rest of the newspaper behind for others to enjoy. Unfortunately, because of the way that paper was folded, not only would the page with that article be gone, so would the page on the reverse side and also two other pages that were physically on the same sheet of folded paper.

For this problem we assume the classic approach is used for folding paper to make a booklet that has a number of pages that is a multiple of four. As an example, a newspaper with 12 pages would be made of three sheets of paper (see figure below). One sheet would have pages 1 and 12 printed on one side, and pages 2 and 11 printed on the other. Another piece of paper would have pages 3 and 10 printed on one side and 4 and 9 printed on the other. The third sheet would have pages 5, 6, 7, and 8.



When one numbered page is taken from the newspaper, the question is what other pages disappear.

Input: Each test case will be described with two integers N and P, on a line, where $4 \leq N \leq 1000$ is a multiple of four that designates the length of the newspaper in terms of numbered pages, and $1 \leq P \leq N$ is a page that has been taken. The end of the input is designated by a line containing only the value 0.

Output: For each case, output, in increasing order, the page numbers for the other three pages that will be missing.

Example input:	Example output:
12 2 12 9 8 3 0	1 11 12 3 4 10 4 5 6

ACM Mid-Central Programming Competition 2013

New Deal

Filename: `newdeal`

The Problem

Deal or No Deal has had its run on television and FOX has decided that they'd like to come up with a different show, ***New Deal,*** to compete with it. In this show, there are a certain number of boxes, each with a different dollar amount, but the added twist is that some boxes have a "negative" amount of cash meaning that if you pick that box, you actually OWE the ***New Deal*** producers some money. In the show, you can pick any subset of boxes and the amount of money you get is the sum of the money in the boxes, including the negative amounts. Luckily, the show is arranged such that most subsets of boxes you could choose have more positive money than negative money.

The one final twist in the show is that if you can hit a particular dollar amount exactly, you are rewarded with **ten million dollars.**

You decide that you want to get on the show and by writing a computer program you'll be able to win the ten million dollars!

The Input

The first line of the input file will contain a single integer, n, representing the number of test cases. Each test case will be on a line by itself, on the following n lines. The first number on each of these lines will be a single positive integer, k $(0 < k < 100)$, representing the number of boxes used in this test case. This will be followed by k non-zero integers representing the dollar values for each box. You are guaranteed that the sum of the absolute values of the dollar amounts in each test case will not exceed 1000000. The last value on each line will be a positive integer, T, representing that exact dollar amount you are trying to hit.

The Output

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

```
Test case #m: You can hit the target T and win $10M!
Test case #m: You can not hit the target T, sorry!
```

where m is the number of the test case (in between 1 and n, inclusive) and T is the target dollar value given in the test case. The first format is printed if some subset of the values within the corresponding boxes adds up to the target amount for that case. Otherwise, print using the second format.

Sample Input

```
2
5 2 3 4 -1 -2 1
6 1 2 3 4 5 6 30
```

Sample Output

```
Test case #1: You can hit the target 1 and win $10M!
Test case #2: You can not hit the target 30, sorry!
```

H: Roller Coaster

Bessie has gone on a trip, and she's riding a roller coaster! Bessie really likes riding the roller coaster, but unfortunately she often gets dizzy.

The roller coaster has a number of distinct sections that Bessie rides in order. At the beginning of the ride, Bessie's dizziness and fun levels are both at 0. For each section of the roller coaster, Bessie can either keep her eyes open or keep them closed (and must keep them that way for the whole section). If she keeps her eyes open for a section, her total fun increases by a Fun factor for that section, and her dizziness increases by a Dizziness factor for that section. However, if she keeps her eyes closed for the section, her total fun will not change, but her dizziness will decrease by a value that's constant for the entire roller coaster. (Note that her dizziness can never go below 0.)

If, at any point, Bessie's dizziness is above a certain limit, Bessie will get sick. Write a program to find the maximum amount of fun Bessie can have without getting sick.

Input

There will be several test cases in the input. Each test case will begin with a line with three integers:

```
N K L
```

Where **N** (1 ≤ **N** ≤ 1,000) is the number of sections in this particular roller coaster, **K** (1 ≤ **K** ≤ 500) is the amount that Bessie's dizziness level will go down if she keeps her eyes closed on any section of the ride, and **L** (1 ≤ **L** ≤ 300,000) is the limit of dizziness that Bessie can tolerate – if her dizziness ever becomes larger than **L**, Bessie will get sick, and that's not fun!

Each of the next N lines will describe a section of the roller coaster, and will have two integers:

```
F D
```

Where **F** (1 ≤ **F** ≤ 20) is the increase to Bessie's total fun that she'll get if she keeps her eyes open on that section, and **D** (1 ≤ **D** ≤ 500) is the increase to her dizziness level if she keeps her eyes open on that section. The sections will be listed in order. The input will end with a line with three 0s.

Output

For each test case, output a single integer, representing the maximum amount of fun Bessie can have on that roller coaster without exceeding her dizziness limit. Print each integer on its own line with no spaces. Do not print any blank lines between answers.

Sample Input

```
3 1 2
2 1
3 1
5 2
10 5 1
20 2
12 4
3 3
10 6
20 3
19 9
19 7
1 500
15 5
4 2
0 0 0
```

Sample Output

```
7
0
```

Problem H: Sort Me

Source file: `sortme.{c, cpp, java}`
Input file: `sortme.in`

We know the normal alphabetical order of the English alphabet, and we can then sort words or other letter sequences. For instance these words are sorted:

ANTLER
ANY
COW
HILL
HOW
HOWEVER
WHATEVER
ZONE

The standard rules for sorting letter sequences are used:

1. The first letters are in alphabetical order.
2. Among strings with the same prefix, like the prefix AN in ANTLER and ANY, they are ordered by the first character that is different, T or Y here.
3. One whole string may be a prefix of another string, like HOW and HOWEVER. In this case the longer sequence comes after the shorter one.

The Gorellians, at the far end of our galaxy, have discovered various samples of English text from our electronic transmissions, but they did not find the order of our alphabet. Being a very organized and orderly species, they want to have a way of ordering words, even in the strange symbols of English. Hence they must determine their own order. Unfortunately they cannot agree, and every Gorellian year, they argue and settle on a new order.

For instance, if they agree on the alphabetical order

UVWXYZNOPQRSTHIJKLMABCDEFG

then the words above would be sorted as

WHATEVER
ZONE
HOW
HOWEVER
HILL
ANY
ANTLER
COW

The first letters of the words are in *their* alphabetical order. Where words have the same prefix, the first differing letter determines the order, so the order goes ANY, then ANTLER, since Y is before T in *their* choice of alphabet. Still HOWEVER comes after HOW, since HOW is a prefix of HOWEVER.

Dealing with the different alphabetical orders each year by hand (or tentacle) is tedious. Your job is to implement sorting with the English letters in a specified sequence.

Input: The input will contain one or more datasets. Each dataset will start with a line containing an integer *n* and a string *s*, where s is a permutation of the English uppercase alphabet, used as the Gorellians' alphabet in the coming year. The next *n* lines ($1 \le n \le 20$) will each contain one non-empty string of letters. The length of each string will be no more than 30. Following the last dataset is a line containing only 0.

Output: The first line of output of each dataset will contain "year " followed by the number of the dataset, starting from 1. The remaining n lines are the *n* input strings sorted assuming the alphabet has the order in *s*.

Example input:	**Example output:**

<pre>8 UVWXYZNOPQRSTHIJKLMABCDEFG
ANTLER
ANY
COW
HILL
HOW
HOWEVER
WHATEVER
ZONE
5 ZYXWVUTSRQPONMLKJIHGFEDCBA
GO
ALL
ACM
TEAMS
GO
10 ZOTFISENWABCDGHJKLMPQRUVXY
THREE
ONE
NINE
FIVE
SEVEN
ZERO
TWO
FOUR
EIGHT
SIX
0</pre>
	<pre>year 1
	WHATEVER
	ZONE
	HOW
	HOWEVER
	HILL
	ANY
	ANTLER
	COW
	year 2
	TEAMS
	GO
	GO
	ALL
	ACM
	year 3
	ZERO
	ONE
	TWO
	THREE
	FOUR
	FIVE
	SIX
	SEVEN
	EIGHT
	NINE</pre>

ACM Mid-Central Programming Competition 2013

G: Star Rectangles

Gregory loves both astronomy and geometry, so he has been plotting stars on a 2-dimensional map. He wants to make the largest rectangle he can from the stars on his map, using a star at each corner, but there are just too many stars for him to solve this task by hand! You have been referred to Gregory as a programming genius who can solve his problem. Gregory has M maps, each with N different stars at coordinates X and Y. His rectangles are always parallel to the x and y axes.

Input

Input begins with the number of maps M ($M \leq 5$). Each map will begin with N ($1 \leq N \leq 100$), the number of stars, on a line by itself. This is followed by N lines containing the X and Y integer coordinates of the stars, each separated by a space. X and Y will be in the range ($-1000 \leq X, Y \leq 1000$).

Output

For each map, print out the coordinates that comprise the four corners of the largest rectangle in sorted order (sort by X, then sort by Y). If more than one largest rectangle can be found, print the one occurring first in sorted order (sort by X, then sort by Y). Your output should follow the exact format shown in the sample output below. If a rectangle cannot be made, print "NONE" on the output line.

Sample Input

```
2
7
-5 5
5 5
-5 0
5 0
0 0
0 11
5 11
7
-5 5
5 5
-5 0
5 0
0 0
0 10
5 10
```

Sample Output

```
(0,0),(0,11),(5,0),(5,11)
(-5,0),(-5,5),(5,0),(5,5)
```

F: Tadpole Catching

Ashley enjoys walking along the edge of the lake near her home. Ashley often sees tadpoles along the shore of the lake and today has decided that she wants to catch as many tadpoles as possible in one scoop. If Ashley has a net with diameter D and its center is located at coordinates X, Y, what is the maximum number of tadpoles Ashley can catch with her net? If a tadpole is positioned on the edge of the net, the tadpole will land inside the net.

Input

Input begins with the number of test cases T ($T \leq 12$). Each test case will begin with N, the number of tadpoles in the lake ($N \leq 50$), followed by D, the diameter of the net ($D \leq 20$). The next N lines contain the X and Y coordinates of each tadpole, where ($-100 \leq X, Y \leq 100$).

Output

For each test case, print out the maximum number of tadpoles Ashley can catch with her net. Your output should follow the exact format shown in the sample output below.

Sample Input

```
1
6 20
1 0
-10 10
5 -5
-7 7
-7 -7
7 -10
```

Sample Output

```
4
```

Problem A: Welcome Party

Source file: `welcome.{c, cpp, java}`
Input file: `welcome.in`

For many summers, the Agile Crystal Mining company ran an internship program for students. They greatly valued interns' ability to self-organize into teams. So as a get-to-know-you activity during orientation, they asked the interns to form teams such that all members of a given team either have first names beginning with the same letter, or last names beginning with the same letter. To make it interesting, they asked the interns to do this while forming as few teams as possible.

As an example, one year there were six interns: Stephen Cook, Vinton Cerf, Edmund Clarke, Judea Pearl, Shafi Goldwasser, and Silvio Micali. They were able to self-organize into three teams:

- Stephen Cook, Vinton Cerf, and Edmund Clarke (whose last names all begin with C)
- Shafi Goldwasser and Silvio Micali (whose first names begin with S)
- Judea Pearl (not an interesting group, but everyone's first name in this group starts with J)

As a historical note, the company was eventually shut down due to a rather strange (and illegal) hiring practice---they refused to hire any interns whose last names began with the letter S, T, U, V, W, X, Y, or Z. (First names were not subject to such a whim, which was fortunate for our friend Vinton Cerf.)

Input: Each year's group of interns is considered as a separate trial. A trial begins with a line containing a single integer N, such that $1 \le N \le 300$, designating the number of interns that year. Following that are N lines---one for each intern---with a line having a first and last name separated by one space. Names will not have any punctuation, and both the first name and last name will begin with an uppercase letter. In the case of last names, that letter will have an additional constraint that it be in the range from 'A' to 'R' inclusive. The end of the input is designated by a line containing the value 0. There will be at most 20 trials.

Output: For each trial, output a single integer, k, designating the minimum number of teams that were necessary.

Example Input:	Example Output:
6 Stephen Cook Vinton Cerf Edmund Clarke Judea Pearl Shafi Goldwasser Silvio Micali 9 Richard Hamming Marvin Minskey John McCarthy Edsger Dijkstra Donald Knuth Michael Rabin John Backus Robert Floyd Tony Hoare 0	3 6