GLOBUSSOFT
TECHNOLOGY AHEAD OF TIME

Greetings From Globussoft

- Given below are 5 Programming questions, you have to solve any 3 out of 5 questions.
- These 5 questions you can attempt in any technology like C/C++, java, .Net, PHP
- To solve these 3 questions you've max. 3 hours.
- While Solving these questions you are not allowed to use any **Search Engine** like Google, Yahoo, Bing …

All the best for your test

Globussoft

QUESTION – 1

Background

So if you want to survive the Earth's demise and see the galaxy, make sure you are wearing something more substantial than pajamas and a ratty bathrobe, watch the skies for flying saucers, start worrying when all the dolphins on the planet vanish, and keep your eyes peeled for an electronic gizmo with the words "Don't Panic" printed in large friendly letters on the cover.

--The Hitchhiker's Guide to the Galaxy

Problem

Addition,subtration and multiplication is necessary wherever you are in the galaxy.You are to write a program to perform these operations, and you can see 6*7=42 or 67-25=42 or 31+11=42, maybe one of them is the problem of Life, the Universe and Everything.

Please note that the solution may only be submitted in Brainfk, Whitespace and Intercal, other languages like C/C++/Pascal/Java are not allowed!** If you want to use other languages to solve this problem, you may try this one.

Input

Multiple test cases, the number of them T is given in the very first line, T<=99.

Each test case contains one line which has the following form:

num **oper** *num*

where *num* is an integer number between 1 and 99, and **oper** is '+' or '-' or '*' (without quotes).

There's no extra whitespace and leading zero in the input.

Output

For each test case you should output one line which contains the answer without any leading zeros.You may assume this number is always a positive one.

Example

Input:
```
3
6*7
67-25
31+11
```

```
Output:
42
42
42
```

QUESTION – 2

You are given an unweighted, undirected tree. Write a program to output the length of the longest path (from one node to another) in that tree. The length of a path in this case is number of edges we traverse from source to destination.

Input

The first line of the input file contains one integer *N* --- number of nodes in the tree ($0 < N <= 10000$). Next *N*-1 lines contain *N*-1 edges of that tree --- Each line contains a pair (*u*, *v*) means there is an edge between node *u* and node *v* ($1 <= u,v <= N$).

Output

Print the length of the longest path on one line.

Example

```
Input:
3
1 2
2 3

Output:
2
```

QUESTION – 3

The Bogus Corporation distributes salary to its employees in a weird manner. The salary is distributed every **K** days, and instead of same salary for each day, the salary for the ith day is a_i. An ambitious young manager, fresh from *Institute of Mismanagement*, observes that people usually prefer to take leave towards the end of this period of K days, when the workload is higher. Instead of revising each of the a_i's, the manager comes up with a quick fix solution - he redefines the new salary on the ith day as $b_i=a_i+2a_{i-1}+2^2a_{i-2}+2^3a_{i-3}+........+2^{i-1}a_1$. Baba, one of the employees, is in a dire financial crisis, and must accumulate at least **N** rupees at the end of the forthcoming period. Being a lazy worker that he is, he is interested in finding out if attending particular days would guarantee him *exactly* **N** rupees at the end of the period. Can you help Baba?

Input

First line contains a single integer integer **T**, the number of test cases (1<=T<=100). Each test case is described on two lines. First line contains two integers, **N** and **K** ($1<=N<=2^{63}-1$, 1<=K<=50) , the second line contains a space separated list of K integers, the **a_i's** ($1<=a_i<=1000$).

Output

For each test case, output on a single line 1-based indices of the days (separated by a single space) he should attend to ensure a salary of exactly N rupees at the end of the period. The indices should be printed in the sorted order. In case of multiple answers, output any one of them. If there is no answer, print -1.

Example

```
Input:
2
9 3
1 1 2
10 2
2 3
Output:
1 3
-1
```

QUESTION – 4

Every year there is the same problem at Halloween: Each neighbor is only willing to give a certain total number of sweets on that day, no matter how many children call on him, so it may happen that a child will get nothing if it is too late. To avoid conflicts, the children have decided they will put all sweets together and then divide them evenly among themselves. From last year's experience of Halloween they know how many sweets they get from each neighbour. Since they care more about justice than about the number of sweets they get, they want to select a subset of the neighbours to visit, so that in sharing every child receives the same number of sweets. They will not be satisfied if they have any sweets left which cannot be divided.

Your job is to help the children and present a solution.

Input

The input contains several test cases.
The first line of each test case contains two integers **c** and **n** ($1 \le c \le n \le 100000$), the number of children and the number of neighbours, respectively. The next line contains **n** space separated integers $\mathbf{a_1}$ **, ... ,** $\mathbf{a_n}$ ($1 \le a_i \le 100000$), where $\mathbf{a_i}$ represents the number of sweets the children get if they visit neighbour **i**.

The last test case is followed by two zeros.

Output

For each test case output one line with the indices of the neighbours the children should select (here, index **i** corresponds to neighbour **i** who gives a total number of $\mathbf{a_i}$ sweets). If there is no solution where each child gets at least one sweet, print "no sweets" instead. Note that if there are several solutions where each child gets at least one sweet, you may print any of them.

Example

```
Input:
4 5
1 2 3 7 5
3 6
7 11 2 5 13 17
0 0


Output:
3 5
2 3 4
```

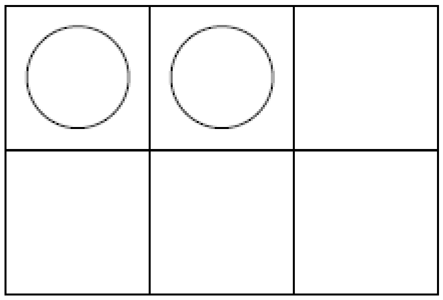
QUESTION – 5

Almost anyone who has ever taken a class in computer science is familiar with the “Game of Life,” John Conway’s cellular automata with extremely simple rules of birth, survival, and death that can give rise to astonishing complexity.

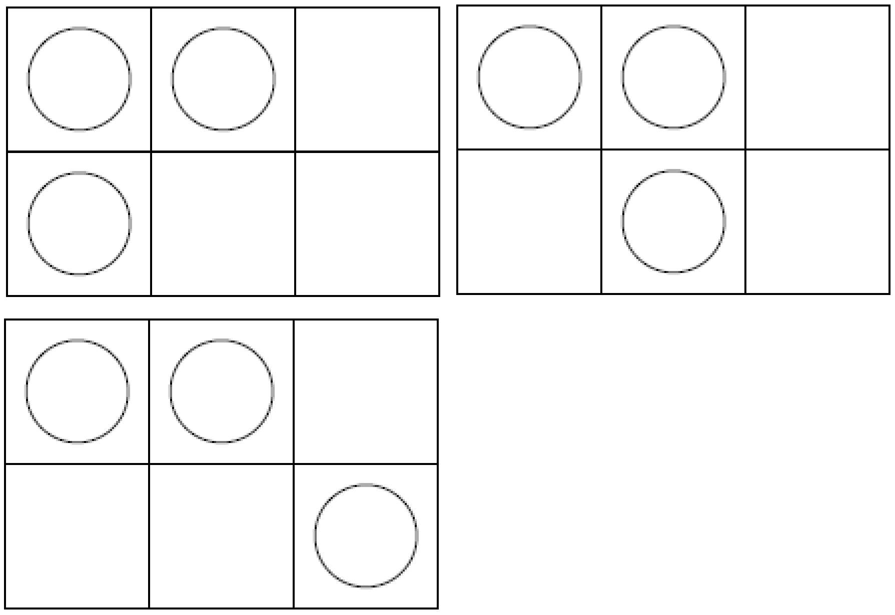
The game is played on a rectangular field of cells, each of which has eight neighbors (adjacent cells). A cell is either occupied or not. The rules for deriving a generation from the previous one are:

- If an occupied cell has 0, 1, 4, 5, 6, 7, or 8 occupied neighbors, the organism dies (0, 1: of loneliness; 4 thru 8: of overcrowding).
- If an occupied cell has two or three occupied neighbors, the organism survives to the next gener- ation.
- If an unoccupied cell has three occupied neighbors, it becomes occupied (a birth occurs).

One of the major problems researchers have looked at over the years is the existence of so-called "Garden of Eden" configurations in the Game of Life — configurations that could not have arisen as the result of the application of the rules to some previous configuration. We're going to extend this question, which we'll call the "Game of Efil": Given a starting configuration, how many possible parent configurations could it have? To make matters easier, we assume a finite grid in which edge and corner cells "wrap around" (i.e., a toroidal surface). For instance, the 2 by 3 configuration:



has exactly three possible parent configurations; they are:



You should note that when counting neighbors of a cell, another cell may be counted as a neighbor more than once, if it touches the given cell on more than one side due to the wrap around. This is the case for the configurations above.

Input

There will be multiple test cases. Each case will start with a line containing a pair of positive integers *m* and *n*, indicating the number of rows and columns of the configuration, respectively. The next line will contain a nonnegative integer *k* indicating the number of "live" cells in the configuration. The following *k* lines each contain the row and column number of one live cell, where row and column numbering both start at zero. The final test case is followed by a line where $m = n = 0$ — this line should not be processed. You may assume that the product of *m* and *n* is no more than 16.

Output

For each test case you should print one line of output containing the case number and the number of possible ancestors. Imitate the sample output below. Note that if there are 0 ancestors, you should print out

`Garden of Eden.`

Example

Input:
```
2 3
2
0 0
0 1
3 3
4
0 0
0 1
0 2
1 1
3 3
5
0 0
1 0
1 2
2 1
2 2
0 0
```

Output:
```
Case 1: 3 possible ancestors.
Case 2: 1 possible ancestors.
Case 3: Garden of Eden.
```