8th Annual Manitoba Programming Contest
September 27, 2014

General Instructions:

1. This document is printed double-sided. Please read both sides of each page.
2. Submit solutions using the PC^2 software.
3. All input should be read from standard input.

UNIVERSITY OF MANITOBA

Problem 1 – Excel-ent Cipher

Ciphers are techniques for encoding text as a secret message. You have been asked to implement a cipher for an accountant friend of yours. Inspired by Microsoft Excel spreadsheets, the cipher works as follows: you are given a number n, which represents the shift of the letters. A shift of 1 means each letter in the original message is replaced by the next letter in the alphabet (e.g., A is replaced by B) and a shift of 4 means each letter is replaced by the letter that four ahead of it in the alphabet (e.g., B is replaced by F).

But what about the letters at the end of the alphabet? That's where the spreadsheets come in. If a shift goes beyond Z, the next letters used are taken from the column labels in Excel: AA, AB, AC, … in that order (after AZ comes BA, for instance). The shifting is guaranteed to never go beyond the column label ZZ.

Write a program that will accept a number (the shift) and a sequence of characters (A-Z) and return the encoded version.

Input

Each line of the input represents a case. The line begins with an integer n satisfying $0 \leq n \leq 676$ that represents the amount of shifting. The line then contains a single space, then a sequence of uppercase letters. There are no lowercase letters, spaces or other characters in the text to be encoded.

Output

For each case, output the encoded version of the string. All characters should be upper case letters (A-Z). Do not include any spaces.

Sample Input	Sample Output
3 CAT 15 DIZZY 1 HAL	FDW SXAOAOAN IBM

Problem 2 – Scor-abble.

You are playing a board game that involves placing tiles that are labeled with letters and scores onto a rectangular grid of positions. For copyright reasons, let's not mention any similarities to any real games.

To score the game, any horizontal or vertical sequence of two or more letters is considered a word. The sum of the score of each tile is counted. If a letter is used twice (both as part of a horizontal and vertical word) then it is scored once for each word. That is, each tile can count in either one or two words, depending on whether it has horizontal or vertical neighbours, or both.

For instance, the following grid represents a four-by-four board:

```
1 1 0 0
1 0 0 3
2 1 1 1
0 0 0 2
```

There are four words in this grid:

- the word of length 2 starting at the upper-left hand corner and going horizontally (score 2)
- the word of length 3 starting at the upper-left hand corner and going vertically (score 4)
- the word of length 4 filling the third row from the top (score 5)
- the word of length 3 in the right-most column (score 6)

The total score for this board is then 17.

Given a board for this game with tiles (represented by their scores only), what is the total sum of all words on the board? In this version, we are not concerned whether the sequences of letters are valid words, only the score of all the words on the board. You are also not concerned with any other rules of any games that may or may not exist (legal plays, bonus scores, etc.).

Input

Each game board is a square grid. The first integer of input for each board is the dimension of the board n (n<50). The next n lines of input each have n integers. Each integer on a line is separated from each other by one space. Integers are less than 100. A value of zero represents that no tile is placed in that spot of the board. Nonzero values represent the score of a tile in that position.

A zero appears on the final line of the input by itself. It should not be processed as part of an input.

Output

Output the score of all words on the board in the format "Case #X: Y" where X is the case number (starting at 1) and Y is the total score for all words on the board.

Sample Input	Sample Output
3 0 0 0 0 1 1 0 0 0 4 1 1 0 0 1 0 0 3 2 1 1 1 0 0 0 2 0	Case #1: 2 Case #2: 17

Problem 3 – The Steep Part

You have been contacted by a company to help them determine information about their fleet of vehicles. The vehicles are given daily tasks where they drive up a hill from a lower elevation to a summit point. The vehicle records the elevation once per second on the trip.

While going from the lower elevation to the summit, the trips are known to never decrease in elevation: the elevation always either stays the same or increases from one second to the next. Many of the trips contain flat segments, however.

For each trip, find the start and end of the segment that contains all the elevation changes. That is, your start point should be the last second at the same elevation as the beginning of the trip, and the end point should be the first second at the same elevation as the end of the trip.

Input

The first line of the input file contains the number of test cases n. It is an integer with $n<10000$.

The subsequent lines contain the test cases. Each test case will span two or more lines. A test case begins with an integer $m < 2^{30}$, alone on a line of input, which gives the number of observations on the trip (in seconds). The next m integers in the input file (separated from each other by spaces or line breaks) represent the m elevation recordings for the trip. Each elevation reading h is between 0 and 2^{16}, and the readings are nondecreasing. The first integer is the elevation at second 0 and the final integer is at second m-1.

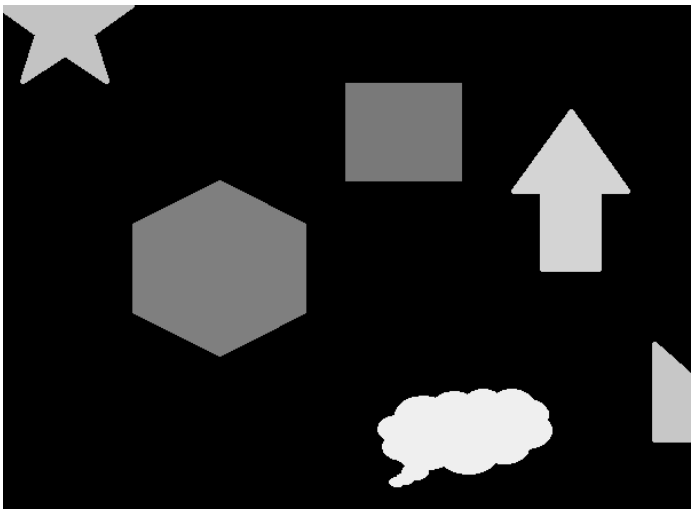
Output

For each input case, give the time points (in seconds, between 0 and m-1 inclusive) representing the shortest interval that contains all the elevation increases for the trip. Format your output as "Trip #X: all changes between seconds Y and Z." where X is an integer starting with 1 for the first case, and Y and Z are the time points.

Sample Input	Sample Output
5 7 0 0 0 1 50 100 100 5 3 3 5 7 7 10 0 1 2 3 4 5 6 7 8 9 10 0 0 0 0 0 0 0 0 0 0 0 0 1 10 0 1 1 1 1 1 1 1 1 1	Trip #1: all changes between seconds 2 and 5. Trip #2: all changes between seconds 1 and 3. Trip #3: all changes between seconds 0 and 9. Trip #4: all changes between seconds 8 and 9. Trip #5: all changes between seconds 0 and 1.

Problem 4 – Object Counter

Peter would like a way to compute the number of objects contained in a greyscale image. Grayscale images contain pixels only coloured with shades of grey. Pixel colours (intensity values) are represented by an integer in the range [0,255], where the 0 and 255 represent the colours black and white, respectively, and values in between are assigned shades of grey. Peter defines objects as *8-connected* pixel clusters, all containing the same intensity value. 8-connectivity is defined below. Here is an example of an image containing 6 objects. The background is always coloured black.



Connectivity is a means to relate pixels in an image based on proximity and grey levels, which is useful for separating components and discovering boundaries. Pixels are considered connected when they are within a certain proximity and have the same grey level value. The 8-connected neighbourhood of a given pixel at position (x,y) is defined as the pixels in positions (x-1,y-1),(x,y-1),(x+1,y-1),(x-1,y),(x+1,y),(x-1,y+1),(x,y+1) and (x+1,y+1). See the figure below.

(x-1,y-1)	(x,y-1)	(x+1,y-1)
(x-1,y)	(x,y)	(x+1,y)
(x-1,y+1)	(x,y+1)	(x+1,y+1)

If there is any pixel in the 8-neighbourhood adjacent to the one at position (x,y) with the same intensity value, then both pixels are considered to have 8-connectivity and belong to the same object.

Input

The input is given as several images. Each image is given by several lines. The first line is the number of rows in the image, the second line is the number of columns, and the remaining lines contain the image pixel intensities, with each row on one line. The number of rows and columns in each image is at most 1000. Each pixel intensity is between 0 and 255, inclusive.

The final two lines of the file each contain only the number 0. These should not be processed as input.

Output

The output for each case should be a single integer representing the number of objects in the image.

Sample Input	Sample Output
3 5 1 0 255 0 35 0 0 0 0 0 0 0 2 2 0 3 3 1 0 2 0 1 0 0 0 0 0 0	4 2

Problem 5 – Camping Bays

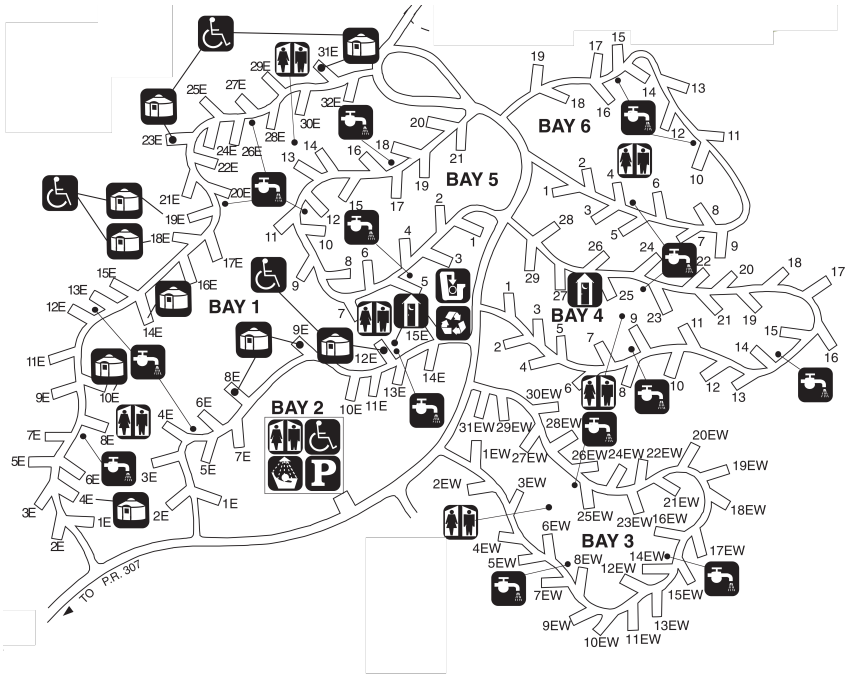
You've been employed at a provincial park campground that has been recently been the victim of pranksters. One of the pranks you've been subjected to is the removal of direction signs in the park. You've found the pile of mixed-up signs and you've been tasked with replacing the signs in the campground.

The part of this task that is giving you the most trouble is replacing the signs on the campground bays. The campground is organized into several bays, each of which has several campsites in it. The bays each have a one-way road, so each bay has an entrance and an exit from the main road. You need to reattach the signs that say "Entrance to bay X" or "Exit from bay X" at the point where each bay meets the main road. A bay never crosses another bay, has any branching, or does anything except leave the main road and arrive at the road at the same point. There is only one copy of each entrance and exit sign for each bay. The entrance to the bay should always appear before the exit to the bay when travelling down the main road.

For example, included on the next page is a map of a set of bays (image adapted from the campground for Nutumik Campground, Whiteshell Provincial Park). Notice that as you drive along the road from the south to the north, you encounter (on the left side of the road):

- the entrance to Bay 1
- the entrance to Bay 2
- the exit of Bay 2
- the entrance to Bay 5
- the exit of Bay 5
- the exit of Bay 1

In doing your job, you have reattached the signs. You can't find a map to tell if you have the signs attached in a plausible way. You don't care if the bays are numbered the same as before (i.e., you don't care if what was Bay 1 before is Bay 5 now). But you want to know if the entrance and exit to each bay is numbered with the same number. Given a description of the signs on **one side of the road** as you drive down the main road, does the order of signs correspond to any valid numbering of the bays at their entrance and exit?



Input

The first line of input is an integer, giving the number of test cases in the input. Each test case is on its own line of input. The test case consists of a strings of characters separated by a single space. The strings will either be of the form "Ex" or "ex" where x is an integer between 0 and 100. The string "Ex" represents the sign for the entrance to bay x, while "ex" represents the sign for the exit from bay x. The sequence of strings represents the sequence of signs as they appear on one side of the main road when travelling down the main road through the campground.

Output

For each test case, output the case number (an integer starting from 1 for the first case), followed by one space then either "valid" or "invalid". Do not include a period after each case.

Sample Input	Sample Output
3 E1 E2 e2 E5 e5 e1 E1 E2 e1 e2 E1 E2 e2	1 valid 2 invalid 3 invalid

Problem 6 – I'm a coding lumberjack

You've taken a summer job as a lumberjack, but so you don't get rusty, you've taken your laptop with you so you can code. As you've been working, you decide to write a program to help you be a better lumberjack.

Your job has a daily routine: you get up, code for a bit, get assigned a tree quota and an area to harvest trees, go cut down trees, code more, go to bed. You are given an area with several trees, all with different heights and weights. To avoid injury, you'd like to harvest trees in both nondecreasing height and increasing weight, so you get warmed up before tackling the largest trees.

As an example, given 5 trees as follows

Tree	1	2	3	4	5
Height	3	10	7	4	11
Weight	1	4	6	5	20

you could cut down Trees 1,4,3,5 in that order, as their heights are 3,4,7,11 and their weights are 1,5,6,20. Both sequences are nondecreasing, so this represents four trees you could cut down from your set of trees.

Given your quota (a number of trees) and the tree heights and weights in your area, is it possible to fulfill your quota while always harvesting trees with increasing heights and increasing weights?

Input

Each input case starts with two positive integers on a line. These represent the number of trees in your area (t), and your quota for the day (q). The value of t is at most 10,000. On the next line, there are 2t positive integers. These are organized into t pairs (h,w), which represent the height and weight of one of the t trees. The final line of input is 0 0. This line marks the end of the input and does not represent an input case.

Output

For each input case, output either "ok, can harvest m", where m is the maximum number of trees that can be cut down in the right order, in the case when $m \geq q$, or output "not ok, can only harvest m", where again m is the maximum number of trees, in the case where $m < q$.

Sample Input	Sample Output
5 2 3 1 10 4 7 6 4 5 11 20 3 3 1 20 2 10 3 5 0 0	ok, can harvest 4 not ok, can only harvest 1