UT Dallas High School Programming Contest November 3rd, 2012

- **Time Allowed: three hours.**
- **Each team must use only one computer - one of UTD's in the main lab.**
- **Answer the questions in any order.**
- **Use Java, C/C++ or C#.**
- **Your program source code must be contained in one source file.**
- **Do not use the "package" construct in your Java code.**
- **Your programs must read from the named file specified in the problem header, and must output to System.out.**
- **Do not access the web except to access Java documentation.**
- **Do not use any recording device containing code, other than UTD's computer.**
- **Your solutions must be entirely yours, typed by you during the time allowed for the contest.**
- **As soon as you have solved a problem, submit ONLY the source file via your `PC^2` client.**

Scoring

Equal points will be awarded to each question, even though they may not be equally difficult.

In order to break ties, we will use penalty points. The penalty points for a question will be zero if the question is not answered correctly. If a correct submission occurs for a question at time T minutes, then T penalty points will be added, plus 20 penalty points for each incorrect submission for that question.

`Suzie will be along to help you.`

A Billiards

Input File: A.txt
Runtime limit: 3 seconds

```
Suzie thinks the first problem is easy if you look at it in a certain way.
No bouncing balls at all . . .
```

Cannon or Carom Billiards is as popular in Europe as Pool in the United States. It is played with three balls on a table without any pockets. In this problem there is only one ball.

The Billiards table has horizontal side a inches and vertical side b inches. A ball is launched from the middle of the table. After $s > 0$ seconds the ball returns to the point from which it was launched, having made m bounces off the vertical sides and n bounces off the horizontal sides of the table.

You have to find the launching angle A (measured from the horizontal), which will be between 0 and 90 degrees inclusive, and the initial velocity of the ball.

Assume that the collisions with each side are elastic (no energy loss), and thus the velocity component of the ball parallel to each side remains unchanged. Also, assume the ball has a radius of zero. Remember that, unlike pool tables, billiard tables have no pockets.

Input
Input consists of a sequence of lines, each containing five nonnegative integers separated by whitespace. The five numbers are: a, b, s, m, and n, respectively. All numbers are positive integers not greater than 10000. Input is terminated by a line containing five zeroes. In none of the test data will the ball collide exactly with a corner of the table.

Output

For each input line except the last, output a line containing two real numbers (accurate to two decimal places) separated by a single space. The first number is the measure of the angle A in degrees and the second is the velocity of the ball measured in inches per second, according to the description above.

Sample Input	Sample Output

200 100 1 2 1
200 100 5 3 4
201 132 48 1900 156
0 0 0 0 0
	14.04 412.31
	33.69 144.22
	3.09 7967.81

B Breakers

Input File: B.txt
Runtime Limit: 3 seconds

`Suzie really likes this problem. She suggests that you solve this one first.`

Maybe you are familiar with the following situation. You have plugged in a lot of electrical devices, such as toasters, refrigerators, microwave ovens, computers, stereos, etc, and have them all running. But at the moment when you turn on the TV, the breaker flips and all your stuff stops working. Your stuff has drawn more current than the breaker's rating. Of course this is a great safety feature that prevents houses from burning down, but it is also annoying to have to find a working flashlight and then to climb around in a dark closet full of your junk to reset the breaker.

You need a program that checks before you turn on an electrical device whether the combined power drawn by all running devices exceeds the breaker's rating, or whether it is safe to turn it on.

Input
The input consists of several test cases. Each test case describes a set of electrical devices and gives a sequence of turn on/off operations for these devices.

The first line of each test case contains three integers n, m and c, where n is the number of devices $n, (n \leq 20)$, m the number of operations performed on these devices and c, the breaker's rating (in Amperes). Then n lines follow each containing one positive integer c_i, the consumption (in Amperes) of the i-th device.

This is followed by m lines also containing one integer each, between 1 and n inclusive. They describe a sequence of turn on/turn off operations performed on the devices. For every number, the state of that particular device is toggled, i.e. if it is currently running, it is turned off, and if it is currently turned off, it will be switched on. At the beginning all devices are turned off.

The input will be terminated by a test case starting with $n = m = c = 0$. This test case should not be processed.

Output
For each test case, first output the number of the test case. Then output whether the breaker was tripped during the operation sequence. The breaker will be tripped if the sum of the currents drawn c_i by turned on devices at some point exceeds the breaker's rating c.

If the breaker is not tripped, output the maximal current drawn by turned on devices that occurred during the sequence.

Output a blank line after each test case.

Sample Input	Sample Output
2 2 10	Sequence 1
5	Breaker was tripped.
7	
1	Sequence 2
2	Breaker was not tripped.
3 6 10	Maximal power consumption was 9 amperes.
2	
5	
7	
2	
1	
2	
3	
1	
3	
0 0 0	

C Is She Here Yet?

`Suzie says, "Go Grandma!"`

Input File: C.txt
Runtime limit: 3 seconds

Grandma is driving north from her house in Tada to your house. She is traveling on the freeway. She has no phone.

You are given her time of departure, her speed, and the total distance she has to travel. You are also listening to the radio to get traffic alerts. From these data you have to calculate Grandma's time of arrival. Her total travel time will be less than 10 hours.

Traffic alerts will be of the form:

`Alert at` t`,` x`m north of Tada,` d `min delay`

where t is the time of day that the holdup begins, given in the style hh:mm, x is an integer number of miles, and $d \in [1, 59]$ is in minutes.

Traffic holdups, once started, remain in effect for at least the remainder of Grandma's journey. When Grandma has to wait at a holdup, her car is stationary for the given time, d minutes.

Traffic alerts could be for locations that Grandma has already passed, or locations on the road ahead of her. They will be given in order of increasing start time.

Input

The first line of the input contains a single integer C giving the number of test cases. Then follows C test cases.

The first line of each test case contains an integer giving the distance in miles that Grandma has to travel, Grandma's time of departure in the format hh:mm, and an integer giving the speed of grandma's car in miles per hour (when she isn't sitting at a traffic holdup).

Then comes a line containing the number of traffic alerts A. A lines follow, each giving the details of a traffic holdup.

Output

For each test case, print a single line as follows:

`Grandma's ETA hh:mm`

Sample input	Sample Output
2 120 12:00 60 1 Alert at 12:30, 10m north of Tada, 10 min delay 120 12:00 60 2 Alert at 12:30, 40m north of Tada, 10 min delay Alert at 01:00, 90m north of Tada, 15 min delay	Grandma's ETA 02:00 Grandma's ETA 02:25

D Don't Run!

Input File: D.txt
Runtime limit: 3 seconds

You have to move along a straight line from x to y, by taking steps, where x and y are positive integers. The length of each step must be a nonnegative integer. Each step can be one unit longer, equal to, or one unit shorter than the length of the previous step.

What is the minimum number of steps in order to get from x to y?
The lengths of the first and the last steps must be 1. For example, you can move 3 places in 3 moves, 1, 1, 1, and 5 places in 4 moves, 1, 2, 1, 1.

Input
Input begins with line containing n, the number of test cases. For each test case, a line follows with two integers: $0 \leq x \leq y < 2^{20}$.

Output
For each test case, print a line giving the minimum number of steps to get from x to y.

Sample input	Sample Output

3
45 48
45 49
45 50
	3
	3
	4

`Suzie has an idea of how to solve this one. She builds an array` $dMax$ `where` $dMax[i]$ `is the maximum distance that can be traveled with` i `steps. Here is the beginning of the array:`

i	$dMax[i]$
0	0
1	1
2	2
3	4
4	6
5	9

`Then she can search for a distance in the array and use the corresponding index` i `to solve the problem.`

E Don't Take Me

Input File: E.txt
Runtime Limit: 3 seconds

Suzie isn't very good at chess, but she knows how the pieces move. She knows that in programming contests it's always best to do a few examples on paper before writing a program.

Almost everyone knows the problem of putting eight queens on a chessboard such that no Queen threatens another Queen. We want a program to compute the maximum number of chess pieces of one kind that can be put on a board of a given size such that no piece can take another.

We only want to know how many Knights, Queens or Kings can be placed on one board, such that no piece can take another.

Input

The first line of input contains the number of problems p. Then p lines follow. Each problem specification consists of one character from the following set k, Q, K, (meaning respectively the chess pieces Knight, Queen or King) followed by the integers $m, (4 \leq m \leq 10)$ and $n, (4 \leq n \leq 10)$, meaning the number of rows and the number of columns or the board.

Output

For each problem specification your program should output the maximum number of chess pieces that can be placed on the given board size such that no piece can take another.

Sample Input	Sample Output
2	32
k 8 8	8
Q 8 8	

F The Doctor Is In

Input File: F.txt
Runtime Limit: 3 seconds

Doc Parker is the internist at an assisted living center. The center is arranged as a series of apartment buildings spread along a single straight roadway. He wants to position his surgery in one of the buildings to minimize the average distance that a resident must travel to see the doctor.

You are given the positions of the n apartment buildings as n integers $d_1 < d_2 < \cdots < d_n$ (these are distances measured from the entrance gate). You are also given the corresponding number of residents in each of the n buildings, $c_1, c_2, \cdots, c_n$.

Select the apartment building where the following sum is minimized.

$$\sum_{j=1}^{m}[\text{distance between the surgery and the apartment where resident j lives}]$$

where m is the total number of residents.

Write a program that computes the ideal location of the surgery and then outputs the average distance that must be travelled by a resident visiting the surgery.

Input
The input contains several test cases. Each test case is a description of an assisted living center and starts with a line containing the number of buildings, $n, (3 \leq n \leq 200)$. Following this will be n lines, each containing two integers, d_i and c_i, $(1 \leq d_i \leq 1000), (1 \leq c_i \leq 50)$

The input will end with a case starting with $n = 0$. This case should not be processed.

Output
For each test case, output the index of the building where the surgery should be built. If more than one equally good solution exists, output the one with the smallest index. On the next line output the average distance to the surgery for that test case.

Follow the format indicated below. The average distance value should be printed accurate to 2 figures after the decimal point. Both of these figures should be printed.

Sample Input	Sample Output
7	The surgery should be in building 4
5 8	The average distance to the surgery is 4.81
6 5	
12 2	
19 20	
20 15	
27 5	
30 3	
0	

Suzie thinks that trying every possible location for the surgery will work. She makes a mental note to ask her teacher if there is a better way.

G Frogs

Input File: G.txt
Runtime limit: 3 seconds

In a second-grade test in China, students were asked to solve the following puzzle:

There are six frogs, sitting on six rocks. Three frogs are facing to the right, then there is an empty rock, then there are three frogs facing left. The seven rocks form a straight line. A frog can only jump to an unoccupied rock that is next to his rock, or to an unoccupied rock two rocks away from his when he jumps over another frog.

Here is an illustration:

```
R1  R2  R3  __  L1  L2  L3
```

The R# frogs face right and can only jump to the right, one or two places. The L# frogs face left and can only jump to the left, one or two places. A frog cannot jump over another frog pointing in the same direction as he is. Initially, only R3 or L1 are allowed to jump to the unoccupied rock.

The object of the puzzle is to find a minimal length sequence in which the frogs jump, resulting in the final state below:

```
L1  L2  L3  __  R1  R2  R3
```

Your job is to write a program to solve the puzzle when there are 2N frogs arranged with N right facing frogs on neighboring rocks, then an empty rock, then N left facing frogs on neighboring rocks.

Input

The input begins with an integer P on a line by itself giving the number of problems to solve. Then P lines follow, each containing a single integer N, $(1 \leq N \leq 20)$, giving the number of frogs facing each way (2N frogs total).

Output

For each problem, output a single line giving the minimum number of frog hops to solve the puzzle.

Sample input	Sample Output
2 1 2	3 8

```
Suzie quite likes frogs.  She says, "Solve a few examples on paper first".
```