Problem I
Struts and Springs
Input file: springs.in

Struts and *springs* are devices that determine the way in which rectangular windows on a screen are resized or repositioned when the enclosing window is resized. A window occupies a rectangular region of the screen, and it can enclose other windows to yield a hierarchy of windows. When the outermost window is resized, each of the immediately enclosed windows may change position or size (based on the placement of struts and springs); these changes may then affect the position and/or size of the windows they enclose.

A *strut* is conceptually a fixed length rod placed between the horizontal or vertical edges of a window, or between an edge of a window and the corresponding edge of the immediately enclosing window. When a strut connects the vertical or horizontal edges of a window, then the height or width of that window is fixed. Likewise, when a strut connects an edge of a window to the corresponding edge of the immediately enclosing window, then the distance between those edges is fixed. *Springs*, however, may be compressed or stretched, and may be used in place of struts.

Each window except the outermost has six struts or springs associated with it. One connects the vertical edges of the window, and another connects the horizontal edges of the window. Each of the other four struts or springs connects an edge of the window with the corresponding edge of the enclosing window. The sum of the lengths of the three vertical struts or springs equals the height of the enclosing window; similarly, the sum of the lengths of the three horizontal struts or springs equals the width of the enclosing window. When the enclosing window's width changes, any horizontal springs connected to the window are stretched or compressed in equal proportion, so that the new total length of struts and springs equals the new width. A similar action takes place for a change in the enclosing window's height. If all three vertical or horizontal components are struts, then the top or rightmost strut, respectively, is effectively replaced by a spring.

You must write a program that takes the initial sizes and positions of a set of windows (with one window guaranteed to enclose all others), the placement of struts and springs, and requests to resize the outermost window and then determines the modified size and position of each window for each size request.

Input

Input consists of multiple test cases corresponding to different sets of windows. Each test case begins with a line containing four integers *nwin*, *nresize*, *owidth*, and *oheight*. *nwin* is the number of windows (excluding the outer window which encloses all others), *nresize* is the number of outer window resize requests, and *owidth* and *oheight* are the initial width and height of the outer window.

Each of the next *nwin* lines contains 10 non-negative integers and describes a window. The first two integers give the initial *x* and *y* displacement of the upper left corner of the window with respect to the upper left corner of the outermost window. The next two integers give the initial width and height of the window. Each of the final six integers is either 0 (for a strut) or 1 (for a spring). The first two specify whether a strut or spring connects the vertical and horizontal edges of the window respectively, and the last four specify whether a strut or spring connects the tops, bottoms, left sides and right sides of the window and its immediately enclosing window.

Each of the last *nresize* lines in a test gives a new width and height for the outermost window – a resize operation. For each of these, your program must determine the size and placement of each of the *nwin* inner windows. The test data is such that, after every resizing operation, every strut and spring has a positive integral length, and different window boundaries do not touch. Also, resizing never causes one window to jump inside another.

There are at most 100 windows and 100 resize operations in a test case, and the outermost window's width and height never exceed 1,000,000. The last test case is followed by a line with four zeros.

Output

For each resize operation in a test case, print the test case number (starting with 1) followed by the resize operation number (1, 2, …) on a line by itself. Then on each of the next *nwin* lines, print the window number, position (*x* and *y*) and size (width and height) of each of the inner windows of the test case as a result of resizing the outer window. Windows in each test case are numbered sequentially (1, 2, …) to match their position in the input, and should be output in that order. Follow the format shown in the sample output.

Sample Input	**Output for the Sample Input**

1 1 50 100
10 10 30 10 1 0 0 0 0 0
70 150
2 1 50 100
10 10 30 10 1 0 0 0 0 0
10 80 20 10 1 1 0 0 0 0
70 150
1 2 60 60
10 10 20 30 1 0 1 1 1 1
90 90
120 120
0 0 0 0
	Case 1, resize operation 1:
	    Window 1, x = 10, y = 60, width = 50, height = 10
	Case 2, resize operation 1:
	    Window 1, x = 10, y = 60, width = 50, height = 10
	    Window 2, x = 10, y = 80, width = 40, height = 60
	Case 3, resize operation 1:
	    Window 1, x = 15, y = 20, width = 30, height = 30
	Case 3, resize operation 2:
	    Window 1, x = 20, y = 30, width = 40, height = 30