J Hyper Knights

Input: Standard Input
Output: Standard Output

'Hyper Knights' is a puzzle game played in a 2D board. Though the game may not be too familiar to you, but in 'KnightsLand' it's a very popular game. The best thing of the game is that many people can play the game together competing others. The rules of the games are as follows:

Initially an **m x n** white board is taken and in each cell an integer is written. After that some cells are colored green and some cells are colored red. All the players are sent one copy of the board. Then each of the players starts placing Hyper Knights (like chess knights) in the board for one hour. No player can see others board.
When placing Hyper Knights, each player can use as many Hyper Knights as they want, but the jury accepts a board if the following constraints are fulfilled.

1) In each cell, at most one Hyper Knight can be placed.
2) A Hyper Knight should be placed in each green cell.
3) No Hyper Knight should be placed in the red cells.
4) No two Hyper Knights in the board should be in attacking positions.
5) A board with no Hyper Knights is rejected.


A Hyper Knight

Two Hyper Knights are said to be in attacking positions if

1) their vertical distance is **3** and horizontal distance is **1** or,
2) their vertical distance is **1** and horizontal distance is **3**

The scoring technique is quite simple. For a player's accepted board, if a cell contains a Hyper Knight, then the player is awarded a score same as the integer written in that cell. And for all his placed Hyper Knights, he sums up his scores. The player whose overall score is highest wins the game. If several players tie; all of them are declared as the winners.
Now, you are given a 'Hyper Knights' board as described, you have to find the maximum score a player can get in that board maintaining all the restrictions.

Input

Input starts with an integer **T (≤ 200)** denoting number of cases.

Each case starts with a black line. Next line contains two integers, **m** and **n (1 ≤ m, n ≤ 30)** denoting the board with **m** rows and **n** columns. The rows are numbered from **0** to **m - 1** and the columns are numbered from **0** to **n – 1**. Each of the next **m** lines contains **n** integers, separated by spaces. The **jth** integer in the **ith** line denotes the integer written in the cell in **ith** row and **jth** column. The absolute value of each integer will be less than 10^6.
Next line contains an integer **P** denoting the number of distinct green cells. Each of the next **P** lines contains two integers **i** and **j**, denoting that the cell in **ith** row and **jth** column is green.

Next line contains an integer **Q (Q < m * n)** denoting the number of distinct red cells. Each of the next **Q** lines contains two integers **i** and **j**, denoting that the cell in **i-th** row and **j-th** column is red.

If you place Hyper Knights in all green cells you are guaranteed that no two of them will be in attacking positions. And you may also assume that no cell will be colored both red and green.

Output

For each case, print a line containing the case number and the maximum score a player can make. After that you have to print the lexicographically smallest Hyper Knight placement that forms the maximum score. Print the row and column position of each Hyper Knight in separate lines. See the samples for details.

To check the lexicographical order we first make a list using each Hyper Knight placement as **[a_1, b_1, a_2, b_2 ... a_x, b_x]**, where cell **(a_i, b_i)** contains a Hyper Knight. After that we find the lexicographical order using these lists. For example, **[1, 2, 21, 5]** is smaller than **[1, 11]** and **[1, 1, 12, 13]** is smaller than **[1, 1, 12, 13, 1, 3]**.

Sample Input	Output for Sample Input
2 3 4 2 1 3 1 7 2 1 100 1 2 1 100 1 1 0 4 0 2 0 1 2 1 2 2 2 4 2 1 1 1 1 1 1 1 0 0	Case 1: 110 1 0 1 1 1 2 1 3 Case 2: 7 0 0 0 1 0 2 0 3 1 1 1 2

Warning: Don't use ***cin***, ***cout*** for this problem, use faster i/o methods e.g ***scanf***, ***printf***.

Problemsetter: Jane Alam Jan, Special Thanks: Manzurur Rahman Khan