**Code Jam 2017 - Qualification Round**

# Fashion Show

## Problem

You are about to host a fashion show to show off three new styles of clothing. The show will be held on a stage which is in the most fashionable of all shapes: an **N**-by-**N** grid of cells.

Each cell in the grid can be empty (which we represent with a `.` character) or can contain one fashion model. The models come in three types, depending on the clothing style they are wearing: `+`, `x`, and the super-trendy `o`. A cell with a `+` or `x` model in it adds 1 *style point* to the show. A cell with an `o` model in it adds 2 style points. Empty cells add no style points.

To achieve the maximum artistic effect, there are rules on how models can be placed relative to each other.

- Whenever any two models share a row or column, at least one of the two must be a `+`.
- Whenever any two models share a diagonal of the grid, at least one of the two must be an `x`.

Formally, a model located in row $i_0$ and column $j_0$ and a model located in row $i_1$ and column $j_1$ share a row if and only if $i_0 = i_1$, they share a column if and only if $j_0 = j_1$, and they share a diagonal if and only if $i_0 + j_0 = i_1 + j_1$ or $i_0 - j_0 = i_1 - j_1$.

For example, the following grid is not legal:

```
...
x+o
.+.
```

The middle row has a pair of models (`x` and `o`) that does not include a `+`. The diagonal starting at the `+` in the bottom row and running up to the `o` in the middle row has two models, and neither of them is an `x`.

However, the following grid is legal. No row, column, or diagonal violates the rules.

```
+.x
+x+
o..
```

Your artistic advisor has already placed **M** models in certain cells, following these rules. You are free to place any number (including zero) of additional models of whichever types you like. You may not remove existing models, but you may upgrade as many existing `+` and `x` models into `o` models as you wish, as long as the above rules are not violated.

Your task is to find a legal way of placing and/or upgrading models that earns the maximum possible number of style points.

## Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case begins with one line with two integers **N** and **M**, as described above. Then, **M** more lines follow; the i-th of these lines has a `+`, `x`, or `o` character (the type of the model) and two integers $\mathbf{R_i}$ and

**$C_i$** (the position of the model). The rows of the grid are numbered 1 through **N**, from top to bottom. The columns of the grid are numbered 1 through **N**, from left to right.

## Output

For each test case, first output one line containing `Case #x: y z`, where `x` is the test case number (starting from 1), `y` is the number of style points earned in your arrangement, and `z` is the total number of models you have added and/or substituted in. Then, for each model that you have added or substituted in, output exactly one line in exactly the same format described in the Input section, where the character is the type of the model that you have added or substituted in. These `z` lines can be in any order.

If there are multiple valid answers, you may output any one of them.

## Limits

Time limit: 20 seconds per test set.
Memory limit: 1 GB.
1 ≤ **T** ≤ 100.
1 ≤ **N** ≤ 100.
1 ≤ **$C_i$** ≤ **N**, for all i.
0 ≤ **M** ≤ **$N^2$**.
No two pre-placed models appear in the same cell.
It is guaranteed that the set of pre-placed models follows the rules.

### Small dataset (Test Set 1 - Visible)

**$R_i$** = 1, for all i. (Any models that are pre-placed are in the top row. Note that you may add/replace models in that row and/or add models in other rows.)

### Large dataset (Test Set 2 - Hidden)

1 ≤ **$R_i$** ≤ **N**, for all i.

## Sample

Sample Input

```
3
2 0
1 1
o 1 1
3 4
+ 2 3
+ 2 1
x 3 1
+ 2 2
```

Sample Output

```
Case #1: 4 3
o 2 2
+ 2 1
x 1 1
Case #2: 2 0
Case #3: 6 2
o 2 3
x 1 2
```

The sample output displays one set of answers to the sample cases. Other answers may be possible. Note that the last sample case would not appear in the Small dataset.

In sample case #1, the grid is 2-by-2 and is initially blank. The output corresponds to the following grid. (In these explanations, we will use `.` to denote a blank cell.)

```
x.
+o
```

In sample case #2, the only cell is already occupied by an o model, and it is impossible to add a new model or replace the o model.

In sample case #3, the grid looks like this before you place any models:

```
...
+++
x..
```

The output corresponds to this grid:

```
.x.
++o
x..
```