Storehouse

`store.p`, `store.c`, `store.C`

Advanced Cargo Movement, Ltd. owns a big storehouse in which a lot of various goods is stored. The storehouse has limited number of bays where a cargo can be loaded. Each day, trucks come to the bays, load their cargo (each truck loads just one type of goods) and leave for the shops. In order to speedup the loading, storekeepers move the goods to the bay in advance. Since the exact quantity of cargo to be loaded is not known beforehand, the storekeepers must prepare more goods than needed and the left-over goods is then returned to the store. Thus, it is useful when the next truck coming to the bay loads the same goods as the previous one, because it saves unnecessary movement of cargo. The capacity of trucks is much smaller than the capacity of the bay, therefore, any number of trucks can be served from one bay without reloading, as long as they want to load the same type of goods.

Your task is to organize which type of goods will be prepared in which bay, so that storekeepers must move the unloaded goods back to the storehouse as few times as possible. At the beginning of each day, no goods is prepared in any bay. The goods left in bays at the end of the day is not counted to the number of moves needed.

Input Specification

The first line of the input contains the number of test cases to solve. Each test case starts with the line containing three integer numbers, B, G, N, $1 \leq B \leq 1\,000$, $1 \leq G \leq 1\,000\,000$, $1 \leq N \leq 1\,000\,000$, separated by a single space. B is the number of bays in the storehouse, G the number of types of goods stored in the storehouse and N the number of trucks coming to the storehouse. The test case continues by N lines. On the i-th line, there is a number t_i, $1 \leq t_i \leq G$, specifying the type of goods the i-th truck wants to load.

Output Specification

For each test case, your program should output "`Case` X`:`" on the first line, where X is the case number starting with one. Then N lines follow. For each i, $1 \leq i \leq N$, the i-th line must contain one of the following strings:

- "`NO ACTION`" — when no action needs to be performed before the arrival of the truck i (i.e., the goods i-th truck wants to load is already at some bay), or
- "`LOAD` b g" — when the goods with number g should be moved to the bay b before the i-th truck arrives. The goods currently present at the bay b is automatically moved back to the storehouse.

Each time some truck arrives, the goods it wants to load must be prepared at some bay. Assume that any truck leaves the bay before the next one arrives, i.e., only a single truck is served at a time.

The number of "LOAD" actions should be as small as possible. If there are more solutions with the same number of "LOAD" actions, choose any one you want. Outputs for different test cases should be separated by a blank line.

Sample Input

```
2
2 4 5
1
2
1
4
1
3 3 3
1
3
2
```

Output for Sample Input

```
Case 1:
LOAD 1 1
LOAD 2 2
NO ACTION
LOAD 2 4
NO ACTION

Case 2:
LOAD 1 1
LOAD 2 3
LOAD 3 2
```