In many programming contests, special prizes are given to teams who solved a particular problem first. We call the first accepted solution "First Blood".

It's an interesting idea to set prizes for "Last Blood". Then people won't submit their solutions until the last minute. But this is dangerous: if the solution got "Wrong Answer" or even "Time limit exceeded", it may be too late to correct the solution.

You may argue that once a submission got "Accepted", the team can send it again, but in this problem, we only consider the earliest accepted solution of a team for each problem, so re-sending an accepted solution does NOT help!

Given all the submissions in a contest, your task is to find out the "Last Blood" prizes for each problem.

Input

There is only one test case. The first line contains three integer n, t, m ($5 \leq n \leq 12$, $10 \leq t \leq 100$, $1 \leq m \leq 1000$), the number of problems, teams and submissions. Each of the following m lines describes one submission: $time$ ($0 \leq time \leq 300$), $teamID$ ($1 \ldots t$), $problem$ (`A ...L`) and verdict('`Yes`' or '`No`'). Submissions are sorted in time order. That means for two submissions of the same "$time$" field, the submission that comes later in the input is received later in the contest (maybe only a few seconds later). No two submissions are received in exactly the same time.

Output

For each problem, print the last blood's $time$ and $teamID$.

Sample Input

```
5 10 18
0 2 B No
11 2 B Yes
20 3 A Yes
35 8 E No
40 8 E No
45 7 E No
50 10 A Yes
100 4 A No
120 6 B Yes
160 2 E Yes
180 2 A Yes
210 3 B Yes
240 10 B No
250 10 B Yes
270 2 B Yes
295 8 E Yes
295 7 E Yes
299 10 D Yes
```

Sample Output

```
A 180 2
B 250 10
C - -
D 299 10
E 295 7
```