


# Problem H

## Administrative Difficulties

**Problem ID:**  
administrativeproblems  
**CPU Time limit:** 1 second  
**Memory limit:** 1024 MB

**Source:** Benelux Algorithm Programming Contest (E 2013)

**License:** 

It is difficult for a spy to do his job without a decent car. The Bureau of Administrative Personnel for Cars (BAPC) spy-car rental company has a large collection of cars that spies can use and handles the resulting administration. Using cars obviously costs money: they require gasoline to operate and repairs are needed quite often, because spies tend to cause accidents a little more often than other drivers.

At the end of the year, all spies need to be billed for the usage of cars in the past year. Last week, there was a major crash in the billing system, rendering it unusable. All that could be recovered was a list of all available types of cars and a log of events for the past year. Using this information, the spy-car rental company wants to obtain a list of the costs for car usage for every spy on record. This list can then be used to send out the bills manually.



Every type of car is registered with a catalog price, the cost to pick up the car and the cost of driving that car per kilometer. The event list contains three types of events: pick-ups, returns and accidents. When a spy picks up a car, he or she must pay the pick-up cost for that car. Once the car is returned, the number of kilometers driven in the car is recorded and the spy must pay for these kilometers. If an accident occurred when the spy was using the car, repairs need to be paid for. Every accident is rated with a severity as a percentage. To repair the car, this percentage of the catalog price is billed to the spy who caused the accident. If any billed cost is fractional, it is rounded up to the next integer before being added to the bill.

The list of all available types of cars is complete. However, because of the crash, some events in the recovered event log might be missing. The spy-car rental company does not want to present spies with an inconsistent bill, so you should detect inconsistencies in the log entries for each spy. The following conditions hold for a consistent event log:

- A spy will pick up a car before returning it.
- A spy will always return a car they picked up.
- A spy can use at most one car at a time.
- Accidents can only happen when a spy is using a car.

### Input

On the first line one positive number: the number of test cases, at most 100. After that per test case:

- one line with two space-separated integers  $n$  and  $m$  ( $0 \leq n \leq 500$  and  $0 \leq m \leq 10\,000$ ): the number of types of cars, and the number of events, respectively.
- $n$  lines with a string  $N$  and three integers  $p$ ,  $q$  and  $k$  ( $1 \leq p \leq 100\,000$ ,  $1 \leq q \leq 1\,000$ ,  $1 \leq k \leq 100$ ), all separated by a space: for each type of car, its unique name, its catalog price, its pick-up cost, and its cost per kilometer driven, respectively.
- $m$  lines starting with one integer  $t$  ( $0 \leq t \leq 100\,000$ ), a string  $S$  and one character  $e$ , all separated by a space: the time of the event, the name of the involved spy, and the type of event, followed by:
  - if  $e = 'p'$  (pick-up): a string  $C$ : the name of the type of car picked up.
  - if  $e = 'r'$  (return): an integer  $d$  ( $0 \leq d \leq 1\,000$ ): the distance covered in the car last picked up by spy  $S$ , in kilometers.
  - if  $e = 'a'$  (accident): an integer  $s$  ( $0 \leq s \leq 100$ ): the severity of the accident, in percents.

All names of cars and spies consist of at least 1 and at most 40 lowercase letters. There will be at most 500 unique spy names in each test case. The events are given in chronological order.

### Output

Per test case:

- one line for every spy referenced in any of the events, containing a string and one integer, separated by a space: the name of the spy and his total car cost. If the event log for the spy is inconsistent, the total cost should be replaced by the string "INCONSISTENT". The lines should be sorted alphabetically by the name of the spy.

You do not need to separate the output of consecutive test cases with empty lines.

Sample Input 1

```
1
2 8
bmw 5000 150 10
jaguar 7000 200 25
10 mallery p bmw
15 jb p jaguar
20 jb r 500
35 badluckbrian a 100
50 mallery a 10
55 silva p jaguar
60 mallery r 100
110 silva a 30
```

Sample Output 1

```
badluckbrian INCONSISTENT
jb 12700
mallory 1650
silva INCONSISTENT
```