## Baby Names, v2015

**Released: Saturday, 29 August 2015, 08.00am**

**Due: Saturday, 12 September 2015, 07.59am**

### The Actual Problem Description

Given **N** *distinct* baby name suggestions (each baby name consists of *only* uppercase alphabet characters of no more than **M = 30** characters) and the gender suitability of that name (integer 1 for male or integer 2 for female), tell Steven how many baby names start with a *prefix* that is inside a given query interval [START..END), where START < END, and both are strings.

There are **Q** queries that you have to answer. Use the most efficient technique that you have learned so far.

You just need to implement mainly these three (or more) methods/functions:

1. void AddSuggestion(String babyName, int genderSuitability)
2. Insert babyName and its genderSuitability into a data structure of your choice.
3. void RemoveSuggestion(String babyName)
4. Remove an existing babyName from the data structure of your choice.
5. We guarantee that babyName must have been added via previous call of method AddSuggestion.
6. int Query(String START, String END, int genderPreference)
7. Query your data structure and report the current number of baby names that start with a prefix that is inside the query interval [START..END), depending on parametergenderPreference:
    - If genderPreference = 0, report the number of both male and female baby names.
    - If genderPreference = 1, report the number of male baby names only.
    - If genderPreference = 2, report the number of female baby names only.

Note that AddSuggestion(String babyName, int genderSuitability), RemoveSuggestion(String babyName), and Query(String START, String END, int genderPreference) operations **can be interleaved.**

**Examples:**

Let there be **N = 4** distinct baby names suggestions initially (added via AddSuggestion method): {(JANE, 2), (JOSHUA, 1), (MARIA, 2), (PETER, 1)}.

- Query("PET", "STE", 1) = 1 as we have (PETER, 1).
- Query("PET", "STE", 2) = 0 because although we have PETER within the query range, it is *not* a female baby name.
- Query("JA", "PETI", 0) = 4 as we have all four baby names (JANE, 2), (JOSHUA, 1), (MARIA, 2), (PETER, 1) that satisfy the requirements.
- Query("JA", "PETA", 0) = 3 as we have (JANE, 2), (JOSHUA, 1), (MARIA, 2). Notice that "PETER" is *outside* the query interval ["JA".."PETA") as "PETER" ≥ "PETA".
- Query("JOSH", "PET", 1) = 1 as we have (JOSHUA, 1). Notice that "PETER" is outside the query interval ["JOSH".."PET") as "PETER" ≥ "PET". Remember that the interval is left-closed and right-open.
- Query("JANE", "MARIA", 2) = 1 because "JANE" is a female baby name that has prefix inside the query interval ["JANE".."MARIA"), but "MARIA" is not included as "MARIA" ≥ "MARIA" (actually they are equal). Remember that the interval is right-open.
- Query("JANE", "MARIANA", 2) = 2, now "JANE" and "MARIA" are inside the query interval ["JANE".."MARIANA") as "MARIA" < "MARIANA".

Now, if we remove one existing baby name using RemoveSuggestion("MARIA"), then if we now ask Query("A", "ZZZ", 0), we should have an answer 3.

**Constraints**

Time Limit: 1s.
1 ≤ **N** ≤ 26, 1 ≤ **Q** ≤ 10.
All baby names have distinct first letter.
Both START and END only contains 1 character (the maximum END is thus 'Z').

**Sample Input**

```
1 JANE 2
1 JOSHUA 1
1 NOTMYCHILD 1
3 A Z 0
3 A Z 1
3 A Z 2
2 NOTMYCHILD
3 A Z 0
```

```
3 A Z 1
3 A Z 2
0
```

**Sample Output**

```
3
2
1
2
1
1
```