CS2010 2016/2017 Sem2: PS2 A

Tags & Categories

Tags:

Categories:

Related Tutorials

Task Content

Patient Names, v2017 (Subtask A)

Released: Thursday, 2nd February 2017, 12:00 noon
Due: Thursday, 16th February 2017, 11:59 pm

You are encouraged to work with other students or teaching staffs (inside or outside this module) on solving this problem set. However, you **must** write Java code **by yourself**. In addition, when you write your Java code, you **must** list the names of every collaborator, that is, every other person that you talked to about the problem (even if you only discussed it briefly). This list may include certain posts in CS2010 Facebook group. If you have access to CS2010 files from senior batch (that is, CS2010 problem sets from year 2011-2016), please refrain from looking at their code verbatim or worse... submit your senior's code. Automatic special checks are done especially on the last/hardest subtask to compare older code with this year's version. Any deviation from this policy will be considered as cheating. If the offender is caught beyond reasonable doubt, he/she will be punished severely, including referral to the NUS Board of Discipline.

Background

Any modern day hospital will have a patient particulars system which contains all the particulars and medical history of patients that have ever been to the hospital. In fact, this system is now centralized, meaning it is shared among all the government hospitals in Singapore. Since there can potentially be millions of patient records, the system needs to have an efficient way of searching for patients and also removing and adding patients into the system.

For this PS we will simulate a VERY simplified version of this system where only the name and gender of a patient will be stored, and perform adding/removing and searching operations on the system.

The Actual Problem Description

Given **N** *distinct* patient names (each name consists of *only* uppercase alphabet characters of no more than **M = 30** characters) and the gender of that patient (integer 1 for male or integer 2 for female), report how many patient names start with a *prefix*[1] that is inside a given query interval[2] `[START..END)`, where[3] `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.

The skeleton program PatientNames.java (click to view) that can handle all input/output details is already written for you.

You just need to implement mainly these three (or more)[4] methods/functions:

1. `void AddPatient(String patientName, int gender)`
   Insert record containing `patientName` and the `gender` into a data structure of your choice.
2. `void RemovePatient(String patientName)`
   Remove an existing record containing `patientName` from the data structure of your choice.
   We guarantee that the record containing `patientName` must have been added via previous call of method `AddPatient`.

3. `int Query(String START, String END, int gender)`
   Query your data structure and report the current number of patient names that start with a prefix that is inside the query interval `[START..END)`, depending on parameter `gender`:
   - If `gender = 0`, report the number of both male and female patient names.
   - If `gender = 1`, report the number of male patient names only.
   - If `gender = 2`, report the number of female patient names only.

Note that `AddPatient(String patientName, int gender)`, `RemovePatient(String patientName)`, and `Query(String START, String END, int gender)` operations **can be interleaved**.

Examples:
Let there be **N = 4** distinct patient names initially (added via `AddPatient` 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 name.
- `Query("JA", "PETI", 0) = 4` as we have all four patient 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 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 patient name using `RemovePatient("MARIA")`, then if we now ask `Query("A", "ZZZ", 0)`, we should have an answer `3`.

Subtask A Constraints (30 points)

Time Limit: 1s.

1 ≤ **N** ≤ 26, 1 ≤ **Q** ≤ 10.

All patient 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 NONAME 1
3 A Z 0
3 A Z 1
3 A Z 2
2 NONAME
3 A Z 0
3 A Z 1
3 A Z 2
0
```

Sample Output

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

Generating Test Data

The given sample input/output are for illustration purpose and are not enough to verify the correctness of your solution.

You are encouraged to generate and post additional test data in CS2010 Facebook group.

Please use PatientNamesVerifier.java (click to view) to verify whether your custom-made test data conform with the required specifications.

Problem Author

Dr Steven Halim/Dr Chong Ket Fah
For CS2010/R.

Footnotes

[1]A prefix of a string $\mathbf{T = T_0T_1...T_{n-1}}$ is string $\mathbf{P = T_0T_1...T_{m-1}}$ where $\mathbf{m \leq n}$.

[2]Notice that the interval is left-closed and right-open.

[3]In Java, you can compare two strings using the `compareTo` method.

[4]If needed, update the constructor of class `PatientNames`.

Submission (Course)

Select course: CS2010 (2016/2017 Sem 2) - Data Structures and Algorithms II ▼

Your Files:

SUBMIT (only .java, .c, .cpp and .h extensions allowed)

To submit multiple files, click on the Browse button, then select one or more files. The selected file(s) will be added to the upload queue. You can repeat this step to add more files. Check that you have all the files needed for your submission. Then click on the Submit button to upload your submission.

CodeCrunch

CS2010 2016/2017 Sem2: PS2 B

Tags & Categories

Tags:

Categories:

Related Tutorials

Task Content

Patient Names, v2017 (Subtask B)

The Actual Problem Description

Please refer to Subtask A for the full problem description.

Subtask B Constraints (additional 40 points)

Time Limit: 3s.

1 ≤ **Q** ≤ 20000, 1 ≤ **N** ≤ 20000.

With up to 20000 patient names, there will obviously be several names with the same first letter. Note that some earlier patient names via `AddPatient(String patientName, int gender)` method can also be withdrawn (removed) via `RemovePatient(String patientName)` method subsequently.

Now, `START` and `END` can have more than one characters and the gap between `START` and `END` is a mix between small range (e.g. `["SA".."STR")`, `["PE".."PO")`, etc) and large range (e.g. `["A".."ZZ")`, `["AB".."YZ")`, etc).

Subtask B has a loose time limit setting of 3 seconds so that students can focus on correctness first before continuing with the next Subtasks C and D.

You are encouraged to explore Java TreeMap or TreeSet for this Subtask B.

Sample Input

```
1 JANE 2
1 JOSHUA 1
1 MARIA 2
1 PETER 1
3 PET STE 1
3 PET STE 2
3 JA PETI 0
3 JA PETA 0
3 JOSH PET 1
3 JANE MARIA 2
3 JANE MARIANA 2
2 MARIA
3 A ZZZ 0
0
```

Sample Output

```
1
0
4
3
1
1
2
3
```

Problem Author

Dr Steven Halim/Dr Chong Ket Fah
For CS2010/R.

Submission (Course)

Select course: CS2010 (2016/2017 Sem 2) - Data Structures and Algorithms II ▼

Your Files:

SUBMIT (only .java, .c, .cpp and .h extensions allowed)

To submit multiple files, click on the Browse button, then select one or more files. The selected file(s) will be added to the upload queue. You can repeat this step to add more files. Check that you have all the files needed for your submission. Then click on the Submit button to upload your submission.

CS2010 2016/2017 Sem2: PS2 C

Tags & Categories

Tags:

Categories:

Related Tutorials

Task Content

Patient Names, v2017 (Subtask C)

The Actual Problem Description

Please refer to Subtask A for the full problem description.

Subtask C Constraints (additional 20 points)

Time Limit: 1s.

To get up to 90 points in this PS, you have to solve Subtask C with your own balanced Binary Search Tree. Your Lab TA will verify whether your **last**[1] submission to Subtask C indeed use the proper balanced Binary Search Tree **written by yourself** and award 20 points manually after deadline (or override your score to 0 point otherwise).

To simplify Subtask C a bit, it is guaranteed that there is no `RemovePatient` method in Subtask C test case. That is, you can concentrate on `AddPatient` and `Query` for Subtask C. Notice that in the sample input below that is only valid for Subtask C, there is no command that starts with integer 2 (that implies a call for method `RemovePatient(String patientName)`).

We are aware that this Subtask C can be very frustrating with lots of "`Time Limit Exceeded`" verdicts. Please balance your urge to finish this Subtask C with the assignments from your other modules.

If you encounter stack overflow due to your recursion in Binary Search Tree code running too deep, try running your Java program with: '-Xss8m' flag. Ask your Lab TA if you are not sure about the meaning of this flag.

There is another constraint of 15 000 Bytes for the Java source code limit, i.e. you should not write excessively-long balanced Binary Search Tree implementation. Otherwise you will get "`Program Size Exceeded`" verdict. The rationale is like in written examinations. Lecturer draws empty boxes with certain small size to prevent students from writing essays when the lecturer is simply expecting a paragraph for an answer.

Note: Plagiarism detection is activated for this Subtask C. It is **extremely unlikely** to have two substantially long Java code (up to 15 000 Bytes) written by two different CS2010 students **independently** to have high degree of similarities although both have the same algorithm.

Sample Input

```
1 JANE 2
1 JOSHUA 1
1 MARIA 2
1 PETER 1
3 PET STE 1
3 PET STE 2
3 JA PETI 0
3 JA PETA 0
```

```
3 JOSH PET 1
3 JANE MARIA 2
3 JANE MARIANA 2
3 A ZZZ 0
0
```

Sample Output

```
1
0
4
3
1
1
2
4
```

Problem Author

Dr Steven Halim/Dr Chong Ket Fah
For CS2010/R.

Footnotes

[1]There is no submission limit for CS2010 PSes and many students will likely submit multiple versions of Java code to clear this challenging Subtask C (and also Subtask D). However, for scalability, your Lab TA will *only* grade your **LAST** submission. Therefore, please make sure that your last submission is the one that you want your Lab TA to grade.

Submission (Course)

Select course: CS2010 (2016/2017 Sem 2) - Data Structures and Algorithms II ▼

Your Files:

SUBMIT (only .java, .c, .cpp and .h extensions allowed)

To submit multiple files, click on the Browse button, then select one or more files. The selected file(s) will be added to the upload queue. You can repeat this step to add more files. Check that you have all the files needed for your submission. Then click on the Submit button to upload your submission.

CS2010 2016/2017 Sem2: PS2 D

Tags & Categories

Tags:

Categories:

Related Tutorials

Task Content

Patient Names, v2017 (Subtask D)

The Actual Problem Description

Please refer to Subtask A for the full problem description.

Subtask D Constraints (additional 10 points)

Time Limit: 1s.

To get up to 100 points in this PS, for subtask D in addition to implementing your own balanced Binary Search Tree as you have done for subtask C, you need to implement something that should be possible with heavier usage of VisuAlgo: Now the method `RemovePatient` exists in Subtask D test case.

To ensure that your understanding about this operation is correct, try deleting vertex 7 on this specific example on VisuAlgo.

Note: Plagiarism detection is activated for this Subtask D.

Problem Author

Dr Steven Halim/Dr Chong Ket Fah
For CS2010/R.

Submission (Course)

Select course: CS2010 (2016/2017 Sem 2) - Data Structures and Algorithms II

Your Files:

SUBMIT (only .java, .c, .cpp and .h extensions allowed)

To submit multiple files, click on the Browse button, then select one or more files. The selected file(s) will be added to the upload queue. You can repeat this step to add more files. Check that you have all the files needed for your submission. Then click on the Submit button to upload your submission.