CSE 2010, HW5

Due Tue Mar 26 at the start of your lab section; Submit Server: class = cse2010, assignment = hw5S*x*Individual
Due Tue Mar 26 at the end of your lab section; Submit Server: class = cse2010, assignment = hw5S*x*GroupHelp
x is 14, 23, or j–your merged section number (or j for java).

To help track down a hacker who has compromised multiple user accounts, we would like to display (potentially suspicious) activities during a certain period of time. How would you design an efficient tool for the task?

The goal of HW5 is to manage the activities and allow the user to specify a time range to display the corresponding activities. Also, we would like the user to be able to add and remove activities (e.g. from different sources such as log files from applications, the network, the operating system). To improve efficiency, your implementation uses a Skip List that includes the following operations:

- get(*key*) // if *key* exists, return value associated with *key*; otherwise, return NULL
- put(*key*, *value*) // if *key* doesn't exist, add entry and return NULL; otherwise, replace value and return the old value
- remove(*key*) // if *key* exists, remove entry and return its value; otherwise, return NULL
- ceilingEntry(*key*) // return the entry with the smallest *key* greater than or equal to key; return null if no such entry exists
- floorEntry(*key*) // return the entry with the largest *key* less than or equal to key; return null if no such entry exists
- subMap(*key*1, *key*2) // return all entries with *key* such that $key1 \leq key \leq key2$

Use getRandHeight() in fakeRandHeight.c (FakeRandomHeight in java) for put(*key*, *value*) (to facilitate eaiser debugging and testing) [gcc -o hw5 hw5.c fakeRandHeight.c]. You may rewrite/modify doublyLinkedList.c/h (DoublyLinkedList.java). Program files are on the course website. We will be evaluating your submission on code01.fit.edu; we strongly recommend you to ensure that your submission runs on code01.fit.edu.

Input: Input is from the command-line arguments for hw5.c (HW5.java):

- filename of actions, each line has one of the following actions:
  - DisplayActivity *time*
  - AddActivity *time activity*
  - DeleteActivity *time*
  - DisplayActivitiesBetweenTimes *startTime endTime*
  - DisplayActivitiesFromStartTime *startTime*
  - DisplayActivitiesToEndTime *endTime*
  - DisplayAllActivities
  - PrintSkipList

For simplicity, times are in HHMMSS format (HH is 00-23, MM and SS are 00-59) [leading zeros are optional]. You may assume the times are unique. Sample input is on the course website.

Output: Output goes to the standard output (screen), each line has a result for the corresponding action:

- DisplayActivity *time activity/none*
- AddActivity *time activity* [*existingTimeError*]
- DeleteActivity *time activity/noTimeError*
- DisplayActivitiesBetweenTimes *startTime endTime time*1:*activity*1 ... or *none*
- DisplayActivitiesFromStartTime *startTime time*1:*activity*1 ... or *none*
- DisplayActivitiesToEndTime *endTime time*1:*activity*1 ... or *none*
- DisplayAllActivities *time*1:*activity*1 ... or *none*
- PrintSkipList
  (S*h*) empty
  ...
  (S1) *time*1:*activity*1 ...
  (S0) *time*1:*activity*1 ...

Sample output is on the course website.

Submission: Submit hw5.c (HW5.java) that has the main method and other program files. Submissions for Individual and GroupHelp have the same guidelines as HW1.

Note the late penalty on the syllabus if you submit after the due date and time as specified at the top of the assignment.