CSE 2010, HW5

Due Tue Mar 27 at the start of your lab section; Submit Server: class = cse2010, assignment = hw5SxIndividual
Due Tue Mar 27 at the end of your lab section; Submit Server: class = cse2010, assignment = hw5SxGroupHelp
x is 2 or 3—your section number.

Consider a wearable activity tracker similar to Fitbit that tracks activities related to your health. How would you design a system such that it can efficiently find and display activities during a certain period of time?

The goal of HW5 is to allow the user to specify a time range to display the corresponding activities. Also, we would like the system to be able to add and remove (erroneous) activities as the user performs those activities (such as walking, running, swimming, ...)

To improve efficiency, your implementation uses a SkipListMap class that has at least the following operations:

- get(k) [p. 403; pseudocode is on p. 439]
- put(k, v) [p. 403; pseudocode is on p. 440]
- remove(k) [p. 403; p. 441]
- subMap(k_1, k_2) [p. 428]

Use DoublyLinkedList [p. 135], which you can modify, to implement the SkipListMap class. You may also implement your own DoublyLinkedList. Use FakeRandomHeight for put(k, v) (to facilitate eaiser debugging and testing). Program files for DoublyLinkedList and FakeRandomHeight are on the course website.

Input: The command-line argument for HW5.java is the name of a file, which has one of the following actions on each line:

- AddActivity *time activity*
- RemoveActivity *time*
- GetActivity *time*
- GetActivitiesBetweenTimes *startTime endTime*
- GetActivitiesForOneDay *date*
- GetActivitiesFromEarlierInTheDay *currentTime*
- PrintSkipList

For simplicity, time is an integer in MMDDHHmm format and date is in MMDD format (MM is 01-12, DD is 01-31, HH is 00-23, mm is 00-59) [leading zero in MM is optional]. The timestamps are unique and each timestamp can only have at most one activity. Sample input is on the course website.

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

- AddActivity *time activity* [ExistingActivityError:*existingActivity*]
- RemoveActivity *time activity*/NoActivityError
- GetActivity *time activity*/none
- GetActivitiesBetweenTimes *startTime endTime time*1:*activity*1 ... or none
- GetActivitiesForOneDay *date time*1:*activity*1 ... or *none*
- GetActivitiesFromEarlierInTheDay *currentTime time*1:*activity*1 ... or none
- PrintSkipList
  (Sh) empty
  ...
  (S1) *time*1:*activity*1 ...
  (S0) *time*1:*activity*1 ...

Sample output is on the course website.

Extra Credit (10 more points) To allow more interesting queries, instead of using int/Integer for timestamps in the Skip List, use the GregorianCalendar class [1] (assume year is 2018), which extends the Calendar class [2]. The time/date format for input and output remains the same. Also, allow these *additional* input actions:

- GetActivitiesForYesterday *currentTime*

[1]https://docs.oracle.com/javase/8/docs/api/java/util/GregorianCalendar.html
[2]https://docs.oracle.com/javase/8/docs/api/java/util/Calendar.html

- GetActivitiesForLastWeekday *currentTime Mon/.../Sun*
- GetActivitiesForLastWeekend *currentTime*

and output:

- GetActivitiesForYesterday *currentTime time*1:*activity*1 ... or none
- GetActivitiesForLastWeekday *currentTime Mon/.../Sun time*1:*activity*1 ... or none
- GetActivitiesForLastWeekend *currentTime time*1:*activity*1 ... or none

If *currentTime* is during a weekend (Sat/Sun), last weekend is the previous weekend.

Submission: Submit HW5.java that has the main method, SkipListMap.java, (modified) DoublyLinkedList.java, FakeRandomHeight.java and other program files. Submissions for Individual and GroupHelp have the same guidelines as HW1.

For Extra Credit, submit HW5Extra.java, SkipListMapExtra.java, and related program files.

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