Assignment 3

The objective of this assignment is to learn and apply use of Java's ArrayList class to user defined objects.

The set of classes will include Tweet, TweetDB, and a Tester class.

Description:

Tweet Class

Tweet class would implement comparable interface. Therefore, tweet objects can be sorted based on Date-time. The Tweet class stores information about the tweet. Tweet class has following data members:

- private String userID; // stores the user id
- private Date date; // convert the date-time string to a java.util.Date format
- private String tweet; // stores the tweeted message

The Tweet class should have the following constructor:

- public Tweet (String userID, String dateTime, String tweet)
  This constructor takes three strings for user id, date-time and text. Set the values using mutators. The String dateTime should be converted to a java.util.Date.

The Tweet class must implement the following methods:

- Mutator an accessor for every data member. Check appropriate exception here.
- public String toString ()
  This method returns a string containing user id, date of tweet and tweet text.
- public boolean equals(Object other)
  For this assignment, consider two tweets are equal if the tweeted text are equal
- public int compareTo(Tweet o)
  Override compareTo() method to compare two tweet objects based on their Date-time.

TweetDB Class

The TwitterDB class keeps a database of tweets. This class should have the following data members:

- private String comment; // describes the contents of the tweets database
- private ArrayList<Tweet> tweets; // stores all the tweets

TwitterDB class should have the following constructor:

- public TwitterDB(String comment)
  The constructor sets the comment instance variable. It also creates an empty ArrayList object tweets.

TwitterDB class should implement the following methods:

- Accessor and mutator for comment data member.
- public void addTweet(Tweet tweet)
  This method adds the tweet passed as the argument to the tweets ArrayList.
- public int getNumberOfTweets()
  This method returns the number of tweets stored in the tweets ArrayList.
- public Tweet getTweet(int i)
  Returns the i^{th} tweet in the tweets ArrayList.
- public ArrayList<Tweet> getSortedTweet()
  Sort the tweets based on Date-time. Returns the sorted tweet list.
  [Hints: Use Collections.sort() method to sort the tweets. For ref. check Chapter 20.6 of textbook. We will also cover it during Session 6.1]
- public Boolean isAlreadyStored(Tweet t)
  This method checks whether Tweet t is already in the list.
  [Hints: equals() method would be used to see if the given tweet is equal to any of the tweet in the list in term of tweet text]
- public String toString()
  Returns the String representation of the tweet database. (Text along with user id and Date-time for all tweets)

Tester Class

Class to test and use your Tweeter class and TwitterDB class.

You should create a twitter data base and add tweet information in it (take tweet information from user). Tell user about the format of Date-time string (Hints: the format you would use while parsing String to Date-time using SimpleDateFormat class method).

Check the size of list.

Sort the list based on Date-time and print the sorted list.

Get and print the tweet at a specific index.

Check whether a tweet is already in the list.

Print the list.

All three classes must include complete and correct JAVADOC comments. You do not have to actually generate your JAVADOCs, you only need to write the comment code. You must use the techniques shown in class and use proper spelling and grammar.

Submission

You must submit :

1. The NetBeans project that contains your assignment, in a ZIP/RAR file.
2. A text (.txt) document containing all of your source code for all three classes.
3. The name of the Zip file should be YourLoginName_Assignment3.ZIP

Evaluation

Your submission will be evaluated based on the following criteria:

Criteria

Functionality
Data members, constructors, accessors /mutators and other methods are written as per mentioned requirements and are functioning as specified. Checking all the mentioned functionalities from the tester class.

Code Efficiency
Program logic is written concisely and is not cluttered with unnecessary tasks or statements; program structures are correct and done in the most efficient way possible. Appropriate variable names used and proper data types used. Method syntax is correct and methods are coded according to specifications. Uses concepts and techniques discussed in class.

Programming Style:
Code uses proper indentation and spacing. Use of comments/documentation. Adheres to programming standards.

Misc.
All other instructions were followed, including submission instructions above. Techniques discussed in class have been used.