## Lab 3

### Preliminaries

The name of your Eclipse project should be abc123-lab3, where you replace abc123 with your abc123 id. The name of the file that contains your main method should be Lab3.java. You will also need a Date.java file and a DateRange.java for constructing and processing Date and DateRange objects. To submit the project, export the project and upload the zip file to Blackboard.

### Task

The task is to read a file, which consists of one date on each line. Each date should be stored in a Date object. If the next date is later than the previous date, a DateRange object should be created from the two dates. All the Date and DateRange object should be printed out.

### Input and Output

Consider the following input:

June 17, 1997

July 23, 1997

September 28, 1980

September 31, 1980

Mar. 2, 1980

Apr. 2, 1980

May 3, 1980

Nov 25, 1989

Dec 25, 1989

Jan 3, 1973

Note that the months might be written in different ways. Also, note that there is one invalid date. This data should result in the following output:

Date: June 17, 1997

Date: July 23, 1997

DateRange: Date: June 17, 1997 - Date: July 23, 1997

Date: September 28, 1980

Invalid Date

Date: March 2, 1980

Date: April 2, 1980

DateRange: Date: March 2, 1980 - Date: April 2, 1980

Date: May 3, 1980

DateRange: Date: April 2, 1980 - Date: May 3, 1980

Date: November 25, 1989

DateRange: Date: May 3, 1980 - Date: November 25, 1989

Date: December 25, 1989

DateRange: Date: November 25, 1989 - Date: December 25, 1989

Date: January 3, 1973

Each of these lines should be the result of print statements like:

System.out.println(date);

System.out.println(daterange);

where date and daterange are Date and DateRange objects, respectively.

### Reading in the Data

Assume that the name of the file is dates.txt. In Eclipse, the file needs to be in the top directory of the project. Reading and printing all the dates and date ranges in the file should be accomplished by a sequence of code like:

Scanner in = null;

try {

in = new Scanner(new File("dates.txt"));

} catch (FileNotFoundException exception) {

System.err.println("failed to open dates.txt");

System.exit(1);

}

while (in.hasNextLine()) {

String line = in.nextLine();

Date date = new Date(line);

System.out.println(date);

// need more code for DateRange objects

}

Note that the code in the Date object should take care of parsing the line. The Date class needs a toString function that returns a String with the correct information.

To test whether one Date is before or equal to another Date, the Date class should have a compareTo function. The Date class should start with a line like:

public class Date implements Comparable<Date> {

The header for the compareTo method should have a header like:

public int compareTo (Date other) {

This method should return -1 if this Date is before the other Date, 0 if they are equal, and +1 if this Date is after the other Date.

### Comments

Create javadoc comments for your Java classes and your methods.

### UML Class Diagram

Create a UML class diagram similar to that shown in the [Chapter 8 notes](http://cs.utsa.edu/~cs3443/notes/ch08.html) for this lab. This class diagram should include all the methods that you implement.

### Rubric

* An incorrect submission will possibly get zero points.
* (80 pts.) The output is correct (no extraneous output) and produced by Strings coming from toString methods.
* (10 pts.) There are javadoc comments: at least one for the class and another for the main method.
* (10 pts.) A UML class diagram is included that has all the classes and the methods listed.

Revision Date Tue Aug 19 2014 11:21:58 GMT-0500 (Central Daylight Time)