CSCI5134 Concurrent Programming and Software Modeling

Fall 2015

Programming Assignment No. 3

Due: Thu. Nov 19th, 2015

**Part A**.

1. Use (and fix) class Time2 from Fig. 8.5 of Chapter 8 PPT to create a jar library file that can be imported into a Java application (you can cut and paste the code from the book example).
2. Create a simple NetBeans Java application project (based on/using class Time2Test in Fig. 8.6) that will use the library package created in a).
3. Create a new class Time2 based on Fig. 8.5 but representing the time internally as the number of seconds since midnight rather than three integer values for hour, minutes and seconds and create a second jar file with your new implementation of class Time2.
4. Show that the same application created in b) can use either package (the one created in a) or the one created in c) without any change to the application. Turn in source code and schedule a demo with our TA the week the program is due to show a test of your program running with a) the original class and b) with the modified class, both should produce the same results. You should run **THE SAME TEST PROGRAM** (b) twice, once using each implementations of Time2 (the old and the new).

**Part B**.

1. Modify one of the classes (original Time2 or your new class produced in part A.c) to include two overloaded methods that can add time to the time objects as follows:
2. one method will add time received as three integers representing number of hours, minutes and seconds to be added

public void addtime(int h, int m, int s) { ... }

1. another one that adds time received as another Time2 object.

Public void addtime( Time2 atime ) { … }

1. Modify the test program from part A.b) to show that time is added correctly. What should your classes do when adding time takes the time over the 23:59;59 max time in a day? Turn in your NetBeans projects and do a demo showing the modified test program running with the modified class.

**Part C**. Write a subclass **time2tz** based on **either** one of your modified classes (from part B) which allows you to maintain the time zone in addition to the time. Allowed values for time zone include: EST, CST, MST, PST. You will need to override the toString and the toUniversalString methods to return the time with its corresponding time zone. Your class should include at least 2 constructors (your choice). Turn in your source code and do a demo showing your testing of the class where you create several time objects and several “time zoned” objects and show the polymorphic behavior of either the toString or the toUniversalString methods.

Your programs should follow good software engineering practices, i.e., **all constructors should call the fully parameterized constructor, constructors in the subclass should call the superclass constructor appropriate to initialize the superclass state, use of get and set methods, overriding methods should call the superclass version of the method to do partial work and OF COURSE use of private fields AND ONLY private fields in your classes**

Develop all your libraries and main program projects in the same folder. Zip that folder and upload it to blackboard on the due dates before midnight.