Assignment 13

(Java Programming Practices)

(Submitted in **Lab on Wednesday**, 4/21/2010 for **10%bonus**)

(Submitted by **1:40pm on Friday,** 4/23/2010 **without penalty**)

Your name:	Score:

Important Notice:

1. Have the Instructor check your submission by the **due time**:
   - If you ***cannot*** complete by due time, you need to submit your project by **email** (attach your project in a ***single compressed file with meaningful names*** for your projects and file(s)).
   - Late-submission ***penalty*** will be applied for any submissions after the due time.
2. The full score for this assignment is 100 points, **15%** of which go to your J**ava code programming style** (if you have followed the Java Code Convention).

You need to use your **textbook (8th edition)** for this assignment (if you *don't* have the textbook, you may use *the slides on the course website*)—when you have completed the assignment, have the instructor check your projects and sign on the assignment papers, and return them to the instructor (***No email*** submissions are required).

1. Practice with Fig. 8.3, and Fig. 8.4 (not necessarily creating NetBeans projects) and pay attention to the Yellowish-shadowed part of the code and learn how to use "***this***" reference.

2. Practice with sample code in Fig. 8.1 and Fig. 8.2:

    a. Create a NetBeans project for this task.

    b. Read the sample code thoroughly till you understand the design and implementation completely.

    c. Draw a UML class diagram for the class *Time1* in the following space:

d. By using Fig. 8.2 (the class *Time1Test* code and the *Output*), ***without referring to Fig. 8.1***, implement t class Time1 based on the UML diagram you just drawn above in the following space (i.e,, handwrite the Java code for the class Time1), then compare your code with the Fig. 8.1, and make necessary corrections.

3. Practice with sample code in Fig. 8.5 and Fig. 8.6:
    a. Create a NetBeans project for this task.
    b. Read the sample code thoroughly till you understand the design and implementation completely.
    c. Draw a UML class diagram for the class *Time2* in the following space:

d. By using Fig. 8.6 (the class *Time2Test* code and the *Output*), ***without referring to Fig. 8.5***, implement t class Time2 based on the UML diagram you just drawn above in the following space (i.e,, handwrite the Java code for the class Time1), then compare your code with the Fig. 8.5, and make necessary corrections.