Assignment 10
(Due date: Friday, 5/8/2009 via email)

Your name:	Score:

This assignment will be used to test your capabilities of writing not only "correct" Java code but also "proper" code (e.g., variable/method naming, how many methods/variables will be used for each class, proper indentation, proper blank line(s), proper commenting, …in one word, producing a correct result does NOT mean a full credit for the assignment! How to submit:

- **Create a proper name for this assignments with subfolders for the projects**
- **Copy your work to the subfolders,**
- **Compress the folder into a ZIP file and send by email to the instructor at byi@salemstate.edu.**
- **You need to double check your work before compressing and sending it!**

For each of the following projects, you need to

1. Create at least 2 classes: one for testing, another one for the major task (we have done so much practice…).
2. Use UML class diagram to design the class for the major task (follow UML rules for the class name, instance variables, methods…).
3. Create at least **3** constructors for the task class.
4. Create at least **3** instances in the testing class with different constructors.
5. Print out **both the testing processes and results** (you need to follow the textbook examples on how to produce the process and results!—to give sufficient information to the users of your program!)
6. **Commenting your code**.

1. **Rectangle Class:** Create a class **Rectangle**. The class has attributes length and width, each of which defaults to 3.0. It has methods that calculate the perimeter and the area of the rectangle. It has set and get methods for both length and width. The set methods should verify that length and width are each floating-point numbers larger than 1.0 and less than 30.0. Write a testing class program to test class Rectangle (the program should prompt the user to provide which inputs and then display the corresponding outputs for the 3 different rectangles).

2. **Savings Account Class:** Create class **SavingsAccount**. Use a **static** variable to store the annual interest rate for all account holders. Each object of the class contains a private instance variable indicating the amount the saver currently has on deposit. Provide method to calculate the monthly interest—this interest should be added to the balance. Provide a **static** method that update the annual interest to a new value. Write a program to test this class SavingsAccount (*with different initial balances/interest rate, display the initial values, then update/input a new interest rate and print the new balances for different save-times, e.g., 6 months, 1 and half years, for 3 different savings accounts*—the program should prompt the user to provide which inputs and then display the corresponding outputs).