Lab Four Assignment (Ch10-14, CS170 – Spring)

Due: 10:00 pm, Wednesday (5/8)

Submission requirement: your lab must be completed as a project in Eclipse with required documentation for each source code by following the steps explained in **Steps to make a zipped Eclipse project file** below this lab spec. Submit it via Assignments link in Canvas. There is an explanation how to submit your lab after you click on the lab assignment specification link in Canvas.

Source code documentation requirement:

- Use of meaningful names for variables, classes, and methods and meaningful comments.
- Your name (last name, first name), class title and section #, the assignment #, and a brief description of the lab as comment lines must show up at the top of each source code.
- There will be no point if your program does not run. There will be points off if you don't follow the instruction and requirements to code your lab.

Steps to make a zipped Eclipse project file

1. Create your lab as a Java project and all of source code should be in this project.
2. Highlight the project title and make right mouse click, select **Properties**, and you will see the directory or folder in your computer where the project is saved.
3. Navigate to that directory, copy that folder to a different directory, say **C:/Temp**, and click on the right mouse button, select **Sent to**, and select the **Compressed/zipped folder**. Your file is ready for submission.
4. For test if your file can be opened in Eclipse, select **File**, then **Switch Workspace**, and select **Other…**, type a new directory as your new workspace for test, say, **Desktop/MyLabs,** Eclipse will create a new workspace. Select **File**, **Import…**, click on **General**, then **Existing Project into Workspace**, and click on **Next**, click on **Select archive file** button, navigate to the zipped project file, and then click on **Finish**. It should be executable now if your project is correct.

Part I (p. 462): Exercise 7 – your program first at all will ask use to create a friend's contact info including the name, email address, and the mailing address until user entered "n" to stop, and then it will display the options for adding, deleting, sorting, searching, and printing of the contact info of the friends. All data entries must be validated using **Validator** class and you may use or modify any version of Validator class provided in the book examples. The following is the detailed spec:

Create an application that can use the email address to search for a friend's name and mailing address. The following programmer-defined classes should be included in the application:

FriendInfo – class stores friend records including name, email address and mailing address and necessary methods.
MySohu – class creates a collection; carry out adding, deleting, searching, sorting, and printing operations. You may use any **Collection** class including **Map** discussed in the textbook. These

operations must be displayed and processed as a menu using **JOptionPane** or any GUI components. Don't use System.out for output.

Validator – Verify all data entries from the user. You may continue to use the Validator class you have developed or use any version of Validator class discussed in the textbook applying the regular expressions in your code.

MySohuApp – the driver code.

Document all source code as required above.

Part II (p. 501): Exercise 13. Use multithreading and thread synchronization to code an application simulating a bank deposit and withdrawal process. Set up an account containing an initial balance of $1000.00 with two shared threads, **Husband** and **Wife**. When a thread makes a deposit in an account, the other thread must wait. If the account balance is $0.00, no withdrawals are permitted. Assume a deposit transaction requires 0.2 seconds to complete and a withdrawal transaction requires 0.5 seconds to complete. Generate random number 1 as deposit and 2 as withdrawal, respectively. Write a driver class **MultiThreadApp** to test your application, repeat and display the operations 50 times. Document all source code as required above.