

LAB #14

ABSTRACTION WITH INTERFACES

Ted Pawlicki
November 1-2, 2004

REMEMBER TO VOTE ON TUESDAY 11/2

This lab should be completed during the scheduled lab sessions. In order to complete this lab you must demonstrate programs to your lab TA. The technical knowledge on how to accomplish these tasks is found in the assigned reading or the lecture notes, but the lab TA will help as well.

1. Define (in it's own file) the following Interface and use it as described below

```
public interface InputReader {  
    String readLine(String prompt) throws IOException;  
}
```

2. Define (in their own files) two classes, **OptionPaneReader** and **ConsoleReader**, that implement the interface above. An **OptionPaneReader** reads input by displaying input dialogs. The **ConsoleReader** reads from a buffered reader attached to *System.in*.
3. Define (in it's own file) a **TestLab14** class, which contains a main method. The **TestLab14** class also contains a static method that reads and returns a "Complex" number – such as the one you defined in Lab13. The static method should be of the form:

```
public static Complex readComplex(InputReader reader)
```

4. The main method in the **TestLab14** class should do the following:
 1. Build an **OptionPaneReader** and a **ConsoleReader**
 2. Use *readComplex* to get two new **Complex** numbers one from the **OptionPaneReader**, one from the **ConsoleReader**
 3. Print out the complex numbers.

Note: The *main* method should be very short. All the "work" should be done in the classes and the *readComplex* method. The call to the Complex constructor must be in *readComplex*, not in *main*.

Additional Note: If you don't have the code from the Complex class from Lab13 available. It's ok to quickly re-define a simple Complex class from scratch with just a constructor and a toString method. If your code is written properly (with low coupling between classes), it would be a simple matter to substitute in a more complete Complex class without having to modify the code you write in this Lab.

5. Cut and paste your source code and screen shots of running code into your lab document and hand it in. Be sure to demo your running code to your TA so that the TA has a record of "checking you off" as having done this lab.

NOTE: Electronic hand in should be working at this point. Submit your lab work through the homework drop box in WebCT. However, if you encounter any problems, revert to paper based hand in.