CS 320, Spring 2014 — Midterm 2 — Due in class, Friday, May 2nd

- The exam is to be completed individually. Do not discuss any of the questions with any other person.
- Do not use any resources other than your notes, the assigned textbooks, and the resources explicitly linked from the course web page.
- Express your answers in your own words.
- I recommend typing your answers, and using a drawing program or UML modeling program (e.g., Violet UML) to create the class diagram for Question 2. However, *neatly* hand-written and hand-drawn answers are acceptable. **Your name should appear on each page.**
- Turn in a hard copy of your completed exam in class on Friday, May 2nd

Question 1. [10 points] Explain why problem domain classes such as model and controller classes should not depend on user interface classes or other "system" classes.

Question 2. [20 points] Draw a UML class diagram showing the following classes and their relationships.

Classes: Filter, CompositeFilter, EchoFilter, LowPassFilter, Pipe

Relationships:

- CompositeFilter **is a** Filter
- EchoFilter **is a** Filter
- LowPassFilter **is a** Filter
- Filter **has** one or more Pipe objects as input
- Filter **has** one or more Pipe objects as output
- CompositeFilter **has** one or more Filter objects

Question 3. [10 points] What are the advantages of "pay-per-use" software development? Pay-per-use software development is where the customer pays a fee each time the software is used rather than paying a flat fee for unlimited use. Hint: think about how pay-per-use would affect the way the developers would approach the project, and what benefits might result for *both* the customer and the developers.

Question 4. [10 points] Given the following code fragment:

```
Card highCard = getHighCard();
if (highCard != null) {
    score = score + highCard.getScore();
}
String symbol = highCard.getSymbol();
```

A potential bug in this code fragment is that if the `getHighCard()` method returns `null`, a null pointer exception will occur when the `getSymbol()` method is called.

(a) Explain how a unit test could achieve full line coverage for this code fragment without causing the null pointer exception. (100% line coverage means that each line of code is reached when executing the unit test.)

(b) Is is possible for a unit test to achieve 100% branch coverage on this code fragment without triggering the bug? Explain briefly. 100% branch coverage means that every branch (decision point) in the method is exercised in each possible direction.

Question 5. [10 points] In your own words, briefly explain the distinction between an *object* and a *class*.

Question 6. [15 points] Briefly explain the motivation of using an `IDatabase` interface to access the object which implements the persistence operations (the methods that retrieve objects from the database and store objects in the database). Explain the benefits of using this interface vs. referring to a specific class implementing the persistence operations.

Question 7. [10 points] In many of your team projects, I suggested using a class called `DatabaseProvider` as follows:

```
public class DatabaseProvider {
    private static IDatabase theInstance;

    public static void setInstance(IDatabase db) {
        theInstance = db;
    }

    public static IDatabase getInstance() {
        if (theInstance == null) {
            throw new IllegalStateException("No database instance!");
        }
        return theInstance;
    }
}
```

The idea is that:

- All code that needs to perform persistence operations uses `DatabaseProvider.getInstance()` to get the reference to the `IDatabase` object that implements the persistence operations
- Somewhere early in the initialization of the system (*e.g.*, in a servlet context listener in a web application) there is code that creates an object implementing `IDatabase` and passes a reference to it to the `DatabaseProvider.setInstance` method

In your own words, explain the motivations for using this class to manage access to the `IDatabase` object.

Question 8. [15 points] What are the advantages of using an SQL database to store object data rather than storing the data in plain files?