CSC 402 / Final

Design Patterns

One topic we did not discuss in class that is important to the maturity of a programmer is the use of *design patterns* in coding. You have probably been exposed to some of these (like factories) because they are prevalent in Java and you may have dealt with others through work or other means.

Christopher Alexander describes software design patterns as *"Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice"*. Not remarkably elegant, but it does get at the core idea.

The book *Design Patterns: Elements of Reusable Object-Oriented Software* was published in 1994 and it made an attempt to categorize and formalize the ideas that had been bouncing around the community. The four authors became known as the Gang of Four. They split patterns into three categories: Creational, Structural, and Behavioral. Another book of the same time period that discusses patterns is *Code Complete* from Microsoft Press. It is not a book focused on design patterns, however. Other books that may be helpful to you are *Thinking in C++* and *Design Patterns for Dummies*. A good *starting point* is always Wikipedia, of course.

For this essay, you are going to research design patterns and investigate some specifics.

- What are design patterns and why do they exist?
- Describe the three categories of patterns
- Investigate two patterns from each category and write a detailed description of each, focusing on the motivation for creating the pattern (in other words, what problem does it solve and how does it solve it?). Here are some suggestions out of the 23 overall patterns:
  - Creational – Abstract Factory, Builder, Singleton
  - Structural – Adapter, Proxy
  - Behavioral – Chain of Responsibility, Iterator, Mediator, Memento, Observer
- Finally, discuss the Model-View-Controller software pattern, an increasingly popular software model especially on the web, in a similar manner. What is it? What problems does it solve? Why does MVC fit web applications so well? Note that MVC is an architectural pattern that describes an *overall* application, unlike the above design patterns that are used *within* an application.

This essay will be turned in with your final, which is on Dec. 9th at 1:00pm. It must be printed (no handwriting), stapled, and legible. I will not accept it electronically, so make sure you remember to bring your copy with you to the final.