CSE 143 Midterm 1 Topic List – Oct. 20, 2004

Here is a brief checklist of the topics that have been covered this quarter and might appear on Monday's midterm. You are generally responsible for anything that has happened in lecture, sections, or in any of the programming projects.

- Programming as modeling; coupling and cohesion; information hiding
- JavaDoc, toString, other properties of well-written classes
- Inheritance – "is-a" (compared to "has-a", containment)
  - Method overriding, dynamic dispatch
  - Use of super in constructors and in methods
  - Class Object; methods inherited in all classes
  - Abstract classes and methods; abstract vs concrete classes
- Method overloading
  - Difference from overriding
  - Overload resolution – how the appropriate method is picked
- Interfaces
- Interfaces vs abstract classes
  - Tradeoffs
  - Complementary use of both in library design
- Testing and JUnit
  - Purpose of testing, strategies, etc.
  - Defining a unit test using JUnit
- Types
  - Interface types, class types
  - Types of objects that implement interfaces or extend classes
  - Dynamic vs static types
  - Type compatibility
- Packages
  - Import declarations; simple vs full names of classes
  - Package declarations (placing classes in packages)
  - Typical naming conventions (files and folders) for Java source files
- Scope and visibility
  - Nested scopes, including methods and inner classes
  - Visibility: public/private/protected/package
- Graphics and user interfaces – Swing
  - Components and containers – JFrame, JPanel, JButton (know the basic ideas and concepts involved; don't memorize detailed JavaDocs)
  - Adding components to containers; purpose of layout managers
  - Graphics – how `repaint()`/`paintComponent()` works; how Java code interacts with the underlying window manager
- Model/View/Controller (MVC) architecture
  - Roles of MVC components; how information is transmitted; notifications
- Event-driven programming
  - Events and event objects; kinds of information that can be found in an event object
  - Objects that generate events and event listeners
  - How listeners register to handle events – the idea of observers
  - Event interfaces and adapter classes
  - Using nested (inner) classes as observers (mainly how inner classes can access local information in a class); not responsible for anonymous classes or, in particular, their syntax