CS 3331 Final Exam Topics

This exam is comprehensive but mainly covers Chapters 7, 8,
11, and 12 of the textbook.

REMINDERS

This test is closed-notes and closed-book. However, you may
bring 1
page (8.5 X 11) of notes (both sides). Your notes must be
your own,
they must be hand written, and they must be turned in with
your
test. This test is to be done individually, and you are not
to
exchange or share materials with other students during the
test.

If you need more space, use the back of a page. Note when
you do that
on the front.

For diagrams and programs, clarity is important; if your
diagrams or
programs are sloppy and hard to read, you will lose points.
Correct
syntax also makes some difference.

TOPICS

In the following, topics marked with + are more important
than those
marked with -. Topics marked with ++ are almost certain to
be on the
test, in one form or another. In general, skills needed to
solve the
test problems are more or less like those required for the
homework.

Design Patterns

```
  - What are design patterns?
  - What is a generic (or reusable) component? How does it
help when you
    write code?
  + Know to refactor programs and eliminate duplicate code,
e.g., by
    using inheritance or delegation.
  + Know to refactor or generalize existing programs to
come up with
    generic (reusable) components.
 ++ Know the following GoF design patterns:
    Abstract Factory
    Factory Method
    Singleton
    Composite
    Decorator
    Iterator
    Observer
    Strategy
    Template Method
 ++ Know when and how to use the above-mentioned design
patterns.
 ++ Write a program that uses above-mentioned design
patterns.

Object-Oriented Application Frameworks
  + What is an application framework? Name a few examples.
  + Describe some of important characteristics of
frameworks.
  - How do frameworks differ from conventional OO library
classes?
  - How do frameworks differ from design patterns?
  - Know the difference between AWT and Swing.
 ++ Write a GUI application or an applet that uses Swing
components
    (or widgets) such as buttons, labels, combo boxes,
    check boxes, list, scroll bars, text fields, text
areas,
    panels, frames, and dialogs.
  + Know how to layout elements of containers by using such
```

```
    layout managers as border layout, card layout, flow
layout,
    and grid layout.
  + Describe the event handling mechanism of Swing
framework;
    explain underlying concepts such as event, event
source, and
    event listener (or handler) and how they interact with
each
    other.
 ++ Write event handling code for a given event.
  + Use an anonymous inner class, esp. in writing event
    handling code.
  + Understand the collection interfaces such as
Collection,
    Set, List, and Map. What are benefits of defining such
    collection interfaces in addition to providing
implementation
    classes?
  - Explain why several different implementations are
provided for
    the same collection interface, e.g., ArrayList and
LinkedList
    for the List interface?
  + Know how to use various collection interfaces and
classes.
  - Write code that enumerates over a collection by using
    iterators.
  + Understand the concept of ordering, natural ordering,
and
    arbitrary ordering.
 ++ Know to define a natural or arbitrary order among
objects;
    i.e., write a class that implements the interface
Comparable
    or Comparator.
  - Understand the class hierarchy of I/O framework,
    in particular, byte streams and character streams.
  + What are the benefits of using the Decorator pattern in
    implementing the I/O framework?
```

+ Know how to use various I/O streams.

Concurrent Programming

- What are threads?

++ Know how to create and control threads.

+ Know how to make a method (or a block of code) atomic.
+ Write cooperative synchronization code; i.e., implement guarded suspension.

Network Programming

- Know the different roles of server vs. client sockets.

+ Know APIs of sockets and be able to use them to write a simple network program.

- Explain the concept of serialization and deserialization.

Additional topics

+ Streams

++Code Smells (identify and fix)

+ Refactoring