Books
The recommended book for this course is:
*CORE Java 2, Volume II -- Advanced Features*, by Cay S. Horstmann & Gary Cornell, *Prentice-Hall*, 2000, ISBN 0-13-081934-4. (Note that this is the absolute latest version, which became available just a few days before Christmas, 1999. There were earlier versions, but this one is more up to date.)
Volume 1 of the same book is also a useful reference for some topics in this course:
I have found the following books to be useful references (they are listed in no particular order):
(Note that the above two books both contain useful information, and appear not to be strongly related, despite the impression that the latter is merely a newer version of the former.)
Course Outline:
The following is a rough schedule for what will be covered in the course.
Day 1  Introduction
Day 2  Exceptions
Days 3 & 4  I/O, Files, and Streams
Day 5  Serialization & Reflection
Days 6 & 7  Threads
Days 8 & 9  Networking
Day 10  Remote Method Invocation (RMI)
Day 11 & 12  JDBC
Day 13 & 14  Java Beans
Day 15  Final Exam
See Assignment Requirements, below, for what I expect you to submit for each assignment.

Course Assignments
Assignment 1 A Game of TicTacToe
Assignment 2 Pig Latin
Assignment 3 Client/Server TicTacToe
Assignment 4 RMI-based Client/Server TicTacToe
Assignment 5 Video Store Database

Philosophy
I try to instill into students the concept of implementing something for a [possibly unknown] client, not merely for their own use in a course assignment. In my opinion, too many college courses never give students the discipline to implement something realistic; too many times, they merely providing short, simple assignments. I don't think that's sufficient, and I don't think it prepares students for what they will have to do in a real job. In other words, I think that I would do my students an injustice if I merely gave you simple, course-specific problems, and expected you to limit yourselves just what's needed for the immediate assignment at hand.

In the real world, you will be asked to implement programs or classes that are usable by your (or your company's) clients, no matter how those clients choose to use those programs/classes. Many times, you have no idea who these clients will be, nor how they might use what you are writing. So you must do as complete a job as possible, define your interfaces as well as possible, and you will be expected to thoroughly test everything that you write (else you or your employers are likely to get a poor reputation, which usually has negative financial implications for both of you!).

So, remember: Don't think in terms of writing assignments just to satisfy the simple assignment requirements. Instead, think in terms of what your potential clients may do when they try to use your class(es). Once you have absorbed that lesson, and learned to write code that way almost as second nature, then you will be much more employable! If you don't learn that, you're likely to have a bumpy road ahead in whatever programming job(s) you may find.

General Guidance
Unless otherwise specified in the assignment, you may choose to implement the programs either as Java applications or Java applets. You'll find that Java applications are much easier to write, since you don't have to write any HTML to run a Java application (unlike a Java applet), but once you've become used to writing the HTML, it isn't really too different, except for the restrictions placed upon applets by the browser' Security Manager. Also, you can create an application and do the debugging there, and then later convert it to being an applet, if you really need to. In fact, you can write it as both an application and an applet if you wish!

Note: If you are using JDK 1.2 or above (aka Java2), please be aware that, until recently, no mainstream browser supported this version directly. Here are some solutions to this dilemma:

Don't write applets -- write applications instead.

Use JDK 1.1.x, or:
Use the Java Plugin. This is a plugin for your browser (Netscape, Internet Explorer, or Opera) which supports JDK 1.2 or 1.3, depending on which one you install. However, you also must transform your <applet> tags in your HTML file(s) for this to work. The Java Plugin comes with a program that automatically does this transformation for you.

Netscape 6.2 is now available, and supports JDK 1.3
Opera 5.01 is now freely downloadable, and supports JDK 1.3, through the Java plugin

All the assignments are intended to be a major part of the learning process: I believe that learning by doing is the most effective approach, especially in computer programming. I also believe in assignments that are reasonably representative of real-world problems, rather than mere academic exercises. For this reason, several of the assignments may build incrementally towards a goal. I hope that these assignments are not only challenging, but also fun!
**Note:** Please do not wait until I've covered a particular topic before starting an assignment that involves that topic. This is a graduate course, and so I expect you to use resources beyond my class lectures -- books, the Web, etc. If you wait for me to cover a topic in class, you are likely to fall behind in the assignments.

**Note:** I expect you to test your assignments thoroughly. Failure to do a good job of testing will affect the grade for that assignment! This implies that you must supply me with evidence of such testing, such as the results of tests.

It is possible that you will find difficulty in completing all the assignments by the end of the course. Some people appear to take more time than others to grasp certain concepts and to complete a given assignment, so it's hard to determine how long you should expect to take. (Note that I have completed all of the assignments myself!). For this reason, we will need to cooperate so that I can monitor your individual progress. If I see some of you progressing more slowly than average, I will attempt to determine the reason, and provide additional help as necessary.

**Important Note:**
Please do not hesitate to contact me if you find yourself not understanding something in an assignment, or if you are stuck making little or no progress! I can probably clear the mental "logjam" reasonably quickly and get you moving again in the right direction. However, I do expect you to have thought about the problem and that you have tried a number of possible solutions before you give up and contact me. This is in your interest, as well as in mine!

I have found in the past that those students who are willing to contact me are often the ones who do better in the course. In past courses, I have been painfully aware that some students were experiencing problems, but unfortunately they often did not try to contact me for help. This was true even after I repeated several times in class my eagerness and willingness to help! This situation is not only frustrating for those students, but it is also very frustrating to me, because I can usually help them overcome a problem relatively quickly, and prevent them from spending a lot of wasted time.

Asking me for help should not be considered in any way a cause for embarrassment or shame. On the contrary, it should be considered more of an introduction to the real world, where you will be expected to ask for help to solve programming problems. Failure to ask for help in the real world can often reflect negatively on your performance. (Naturally, if you ask for help before having thought about the problem sufficiently, or if get into the habit of asking for help to avoid doing work, then that is a different matter.)

Please note the following:
I have never penalized someone for asking for help (and I have had students who have asked for help many times!) On the contrary, I have found that those who ask for help are much more likely to get a better grade than those who do not.

Students who do not ask for help often penalize themselves by failing to complete assignments, by completing assignments incorrectly, or by misinterpreting what is expected from an assignment.

Note that email is the best way to contact me, because it is more likely to find me than a telephone call, it's a lot faster than the US Postal Service, and you can explain your problem in more detail than you can usually via telephone. I can also cut and paste code you send me to try things out myself.

Assignment Requirements
For each assignment, I expect you to submit the following:
A Microsoft Word document (or equivalent). I prefer hard copy, handed in at class.
The document will contain:
**Your name. (Remember, I have a lot of students to grade, and need all the help I can get to avoid mixing listings up!)**

A description of what you did. (Please don't repeat my assignment words back to me; I've already read them!) The description doesn't have to be long. I'm more interested in quality than quantity!
Listings/Source for all classes that you have written (please do not submit listings for classes that I supply to you, unless you have changed them in some way -- in that case, please highlight the changes you have made). The listings will be in a monospaced font such as Courier New (I suggest you use a
paragraph style such as Plain Text.) -- this is so I can read them easily, and so that tabs can line up properly. Use a small enough point size for the monospaced font, that the listing lines do not wrap. I recommend that you go into your Java editor, and set it so that tabs are converted to spaces; that really helps with lining up and indenting code!

Output from your program, to show that it works correctly. This often involves capturing a graphical image of part of your screen. On Microsoft Windows, there are at least two ways to achieve this:

Use Ctrl/Print Scrn (while holding down the Ctrl key, press the PrintScrn key) to place a bitmap of your screen into the Windows paste buffer. You can then paste the result into Microsoft Paint, or other programs such as Microsoft Word. If you do this, be aware that the resulting bitmap can be quite large. It's a good idea to use a program such as Microsoft Paint to cut the bitmap down to something more manageable than the entire screen. Also, if you sent bitmaps through email, please ZIP them up first to compress them (you do all use WinZip, right?) -- if you include bitmaps in your Word documents and send them through the mail, use one of the utility programs available on the web to capture parts or all of your screen. One of the most popular is SnagIt (which is not freeware, so please do the right thing!) With these programs, it's very easy to specify that you merely want a particular window to be saved, and you can also specify where to save it.

Answers to any questions I ask in the assignment. (I will take off points if you do not answer the questions.) Furthermore, I expect you to:

Follow the standard Java Coding Conventions (see below), including the naming conventions. Provide meaningful comments for every class and every class method (including static methods). By meaningful, I mean that they describe, at an appropriate level, what the class or method does. The comments should not be merely fluff, just to fulfill the requirement, and should not simply repeat something that the reader can determine from reading the code in a few seconds -- that's a waste of both your time and mine. Also, it's usually a good idea to put comments at various points in your method code; typically, a method's code can be organized into sections, and comments at the start of a section are often very helpful, if they describe what the section does at a relatively high level.

Class comments should immediately precede the class statement, and should conform to the javadoc conventions:

```java
/**
 * Some useful comment about the class.
 * @author yourName
 * @version 1.0
 */
```

Method comments should immediately precede the method, and should conform to the javadoc conventions:

```java
/**
 * Some useful comment about the method.
 * @param paramName description of param
 * ...
 * @returns description of return value/type
 */
```

Method section comments should look as follows:

```java
//
// Some useful comment about the section
// of code that follows.
//
```

Variable declarations should have a brief comment of the type:

```java
int myVariable = 53; // Brief description of its use
```

**One more thing:** I think that the practice of sprinkling "magic numbers" throughout your code is very poor.
For example:

```java
myFavoriteMethod("Frodo", 80);
```
or:

```java
flag = 2;
```
are examples of such "magic numbers". There are two problems with doing this:
The meaning of the "magic number" is very often obscure. A comment helps, but isn't really sufficient in all cases.

Often, the number is repeated many times in different parts of code. If, for some reason, the value of such a "magic number" needs to be changed, this forces the programmer to find every occurrence of the value where a change would be appropriate, and eliminate all occurrences of the value where a change would be inappropriate because its meaning is different. For example, if the number 3 were used to represent a count of items in one context, and a flag value in another context, it becomes very hard to maintain your code if there are many cases of hard-coded 3's sprinkled throughout your code.

The solution is to define a constant with a meaningful name, and use that consistently, instead of a "magic number". Then, the above examples might become:

```java
myFavoriteMethod("Frodo", SCREEN_WIDTH);
```
or:

```java
flag = SHOW_WIDGET;
```

Note that the following is NOT a suitable substitution:

```java
flag = TWO;
```
because it fails to convey the meaning of the value.

---

**Java Coding Conventions**

Before you start doing any real coding, read the JavaSoft [Java Coding Conventions](https://docs.oracle.com/javase/tutorial/javacodingconventions/), especially the section on **Naming Conventions**.

It is very important to start coding using the proper naming conventions -- they are not optional when it comes to using Java Beans, etc., so let's start using good habits early!

I will expect you to follow these conventions in all the code you submit to me. There is one *exception*: The use of braces in section 7.2 **Compound Statements**. Their bracing conventions use the following form:

```java
while (!done) {
    statement;
    statement;
}
```
(Actually, if you look at their actual code in the Java classes, it's even less readable than the above!)

I prefer the much more readable form:

```java
while (!done)
{
    statement;
    statement;
}
```

While you may choose to use the JavaSoft (aka K&R) convention in your personal code, I request you to use my preferred form in the code you submit for these assignments; I find the K&R convention much harder to read, and that adds unnecessary effort to my grading your assignments.

**NOTE:** Regardless of brace alignment conventions, I require you to indent statements consistently. Code that does something like:

```java
while (!done)
{
    
```
Is very hard to read, and is often associated with sloppy work. If I encounter this kind of inconsistent formatting, the grade will suffer!

Note: One reason why this happens when you print out your Java sources is that your editor usually has settings for where the tab stops are placed, and when you hit Tab on the keyboard, the editor will take you to the next tab stop that it knows about. Often, this is every 2 or 4 columns. However, when you print the source out, the printer knows nothing about the tab stops and messes up your formatting. To solve this problem, please set your editor's formatting characteristics so that any time you hit Tab, it gets translated into the appropriate number of spaces necessary to get to the next tab stop.