1 Basic Course Information
1.1 Objectives:
This course explores data structures, algorithms for manipulating them, and the practical problems of implementing those structures in real programming languages and environments. Heavy emphasis is placed upon the analysis of algorithms to characterize their worst and average case requirements for running time and memory.

* Copyright 1999-2002, Steven J. Zeil, Old Dominion University
Perhaps more than any other course, CS361 should expand the students “toolbox” of basic techniques for manipulating data at both the conceptual and the concrete level. At the conceptual level, the student will see a broad selection of standard practices and approaches used in program design. At the concrete level, the student will begin what should be a career-long practice of accumulating useful, reusable code units.

1.2 Required Text:


2 Course Prerequisites

- CS 250, Problem Solving and Programming,
- CS 252 (a.k.a. CS 395), Introduction to Unix for Programmers
- MATH 163, Pre-Calculus II,

or equivalents.

Students are presumed to be familiar with basic programming techniques, including the use of pointers, functions and procedures, loops and recursion. Also assumed is facility with basic algebra.

Students are also expected to be familiar with the use of standard Internet-based tools including

- e-mail
- telnet (for logging on to other machines)
- ftp (for transferring files among machines)

2.1 Do I Need to Know C++?

Yes. CS 250 is a prerequisite for this course. That means that I will assume that you are familiar with the basics of C++, including

- the various C++ statements and control-flow constructs,
- the built-in data types,
- the use of arrays and pointers,
- the use and writing of functions, and
- the basic use of structs and records for implementing abstract data types.

In addition, students should be familiar with certain basic programming techniques that are largely independent of any specific programming language:

- software design (i.e., top-down design, also known as “stepwise refinement”)
- testing software, including the use of scaffolding code (stubs and drivers) and the selection of test data for functional testing, special values testing, and boundary value testing.
- debugging, including the use of debugging output, of using automatic debuggers to set breakpoints and trace program execution, and the general process of reasoning backwards from failure locations to the faulty code responsible for the failure.

Students who have taken CS 250 in Spring 2002 or earlier are presumed to have learned this material in the 250 laboratories, but may wish to review some of the material in preparation for this course.
2.2 Do I Need to Know Unix?

All students in the course will receive accounts on the CS Dept. network of Unix workstations, and knowledge of how to work with Unix is part of the course prerequisites (CS 252). This course does not require familiarity with shell scripting. It also does not require use of the X windowing system, though students may find X to be a convenience when developing software. All other topics in CS 252 are required. In particular, students must be comfortable with transferring files to and from remote systems and with developing software on remote Unix systems.

3 Technical Requirements - Hardware

Because this course is hosted on the internet, you will need to make sure that you have access to the appropriate computing equipment and software to participate in the course activities.

Hardware requirements are pretty straightforward.

**Required** Almost any Pentium-class Windows 95, 98, NT, 2000, or XP machine should do. Equivalently powered Unix or Macintosh machines are also acceptable, though these may introduce other limitations regarding software (below).

Whatever the machine, a good Internet connection is essential. You should be able to connect to your Internet Service Provider (ISP) at 28,800k or better. Perhaps more importantly, your ISP must be able to actually feed you information at that rate during the time periods when you will be logged in. It’s not unheard of for people to invest in very fast modems only to find that their net access stays the same - because their ISP was too overloaded to serve up the information as fast as their modem could accept it. Unfortunately, this is not an easy thing for individuals to determine (and few ISP’s will have the candor to admit when they have too many customers for their own good).

4 Technical Requirements - Software

Software requirements are more specific:

**Required:**
- Web browser: Acceptable are Internet Explorer 5.01 or better, and Netscape 6 or better. Older versions of these browsers may also be acceptable, but cannot be guaranteed so as the course materials are not tested with older browsers. The AOL browser is *not* acceptable.
- Java: A Java engine is probably included in your web browser. You may need to activate it from your browser’s "settings" or "preferences" area. Occasionally, an otherwise correctly installed browser will not be able to run Java programs (applets). The Sun Java plugin is *very* strongly recommended for Internet Explorer users.
- e-mail: Any MIME-compliant e-mail program should do.
- telnet, ftp: Any program should do.
- C++ compiler: This gets tricky. C++ is a language that is still undergoing rapid change. An international standard for the language has been adopted only within the last year, and, despite what they may say in their advertising campaigns, *no* compiler yet fully implements that standard. Instead, different compiler vendors are implementing the portions of the standard that they can get to market most quickly, leaving the rest till later. Consequently, code accepted by one C++ compiler may fail to compile in another. The “official” compiler for this course is the Free Software Foundation’s g++ (also known as gcc or GNU CC), version 3.2.3 or higher. This is the compiler that the instructor and/or grader will use in evaluating and grading projects. If you have access to other compilers, you may use them,
but you are responsible for making sure that their projects can be compiled by the instructor and/or the course’s grader using the official compiler.

You may want to develop your programs on the most convenient compiler and then port it over to the official environment. Please don’t underestimate the amount of time that may be involved in coping with subtle differences among compilers.

The g++ compiler can be downloaded for free from a variety sources. In particular, Windows 9x/NT users should look here and may want to consult my notes on its installation and use.

Optional:

- Microsoft Netmeeting: This program, available for free from Microsoft, allows students to participate in internet conferences via audio and/or video or typed "chat". Its big advantage over a conventional internet-chat tool or even over internet-based phone programs is that the participants in a conference can show each other (and even share control of) almost any Windows-based tool. Students can use Netmeeting to meet with the instructor during "office hours", call up their assignment code in an editor, and show it directly to the instructor. If you plan on using Netmeeting, you will may want to have a microphone on your system. A digital video camera would also be nice, but is certainly not required (and might overtax your internet connection).

5 Meeting Times

This is an internet-based course. For the most part, students will work at their own pace, at their own chosen times, subject to weekly deadlines for assignments and quizzes.

There is only one regularly scheduled “meeting”, the optional but recommended orientation session shortly before the start of the semester. The date and time for this meeting will be announced on the course web page.

6 Communications

Steven J. Zeil 248D Education Building
(757) 683-4928 Fax: (757) 683-4900
course-related e-mail: cs361z@cs.odu.edu
other e-mail: zeil@cs.odu.edu

6.1 e-mail:

I will use electronic mail on the CS Dept. network for timely communication, especially of clarification/corrections/changes to homework or projects. Students should check their e-mail on a regular basis.

6.2 Office Hours:

Students may meet with the instructor in person or via internet-conferencing. A week-by-week schedule of available meeting times can be found at http://www.cs.odu.edu/zeil/officehours/

6.3 Computer Access:

All students taking this course must have activated a login and e-mail account on the CS Dept’s Unix network. The account setup and password can be obtained at http://www.cs.odu.edu/ by clicking on “Online Services”, then on “Account Creation”.

Students at the Norfolk campus will have access to the Dept’s local network of Unix workstations and PC’s in Educ 245. Students taking the course elsewhere may connect to the machines on this network via the internet. Many Teletechnet sites, particularly those hosted at the ODU regional centers and at Virginia Community Colleges, will have available equipment consistent with the requirements of this course.
7 Course Policies

7.1 Due Dates:
All due dates listed in the course schedule are “suggestions”, intended to guide students in planning and monitoring their progress through the course. The only hard and fast dates will be at the midterm and the end of the course.

- The midterm exam must be completed by the date specified in the course schedule.
  - All assignments, quizzes, etc., with suggested due dates prior to the midterm exam must be completed before the opening date of the midterm exam.
- The final exam must be completed by the date specified in the course schedule.
  - All assignments, quizzes, etc., must be completed before the opening date of the final exam.

Exceptions to these dates will be made only in situations of unusual and unforeseeable circumstances beyond the student’s control. “I’ve fallen behind and can’t catch up”, “I’m having a busier semester than I expected”, or “I registered for too many classes this semester” are not grounds for an extension.

7.2 Academic Honesty:
Everything turned in for grading in this course must be your own work. The instructor reserves the right to question a student orally or in writing and to use his evaluation of the student’s understanding of the assignment and of the submitted solution as evidence of cheating. Violations will be reported to the Honor Council for consideration for punitive action.

By CS Dept. policy, students found to be in violation of this rule will, at the very least, receive a failing grade in the course and may be subject to stiffer penalties. Students who contribute to violations by sharing their code/designs with others are subject to the same penalty. Students are expected to use standard Unix protection mechanisms (chmod) to keep their assignments from being read by their classmates. Failure to do so will result in grade penalties.

This policy is not intended to prevent students from providing legitimate assistance to one another. Students are encouraged to seek/provide one another aid in learning to use the operating system, in issues pertaining to the programming language, or to general issues relating to the course subject matter. Student discussions should avoid, however, explicit discussion of approaches to solving a particular programming assignment, and under no circumstances should students show one another their code for an ongoing assignment, nor discuss such code in detail.

7.3 Grading:

Assignments & Quizzes: 60%
Midterm Exam: 15%
Final Exam: 25%

Students are expected to complete both exams and at least 75 assignments and quizzes. Students may submit more than that, in which case I will drop a corresponding number of the lowest assignment/quiz grades that they have received.

8 Topics

1. Analysis of Algorithms
2. Abstract Data Types
3. Average-Case Analysis
4. Linear Sequences - Vector and List
5. Sorting
6. Stacks, Queues, and Deques
7. Sets and maps
8. Trees
9. Hashing
10. Heaps and Priority Queues
11. Graphs and NP Problems