Computer Science 1321 Course Syllabus*

Jeffrey D. Oldham

2000 Jan 11

1 Course

Course: Problem Solving and Algorithm Design II

Prerequisites: CS1320 or instructor consent

This course is the second course for computer science majors, following the guidelines established by the Association for Computing Machinery. This course also partially satisfies the requirements for *Understanding the World Through Science* of the common curriculum.

The course content will include defining data types including singly-linked lists, doubly-linked lists, stacks, queues, and trees; recursion; use of libraries; pointers; dynamic memory; type-independent programming; and program implementation strategies.

1.1 Course Goals and Objectives

The objectives of this course include, but are not limited to, the following:

- learning fundamental problem solving methodology
- implementing algorithms using a programming language
- dealing with complex systems
- development and analysis of algorithms
- introduction to the basic topics in data structures

2 Instructor

Instructor: Jeffrey D. Oldham

Email: oldham@cs.trinity.edu (If your Windows email program bounces the email, use a better program or send to Jeffrey.Oldham@trinity.edu.)

WWW: http://www.cs.trinity.edu/~joldham0/1321/

Telephones: 210.999.8139 (office), 210.832.9879 (home)

Office: Halsell 201J

Office Hours: Office hours include two hours in a computer lab. If I have time and there is interest, we may discuss using computer tools during these lab hours.

3 Text

Text: *Data Structures and Other Objects Using C++* by Michael Main and Walter Savitch, Addison-Wesley, 1997. WWW resources and errata are available.

3.1 Additional References

I have asked the library to place some of them on reserve for one-day checkout.

- Cormen, Leiserson, and Rivest, *Introduction to Algorithms*, McGraw Hill, 1990. (emphasizes algorithms, not programming)
- Cygnus, The ISO/ANSI C++ Draft Standard. (if you want to know about some obscure C++ rule and have lots of hours to understand the terminology)
- Daniel P. Friedman and Matthias Felleisen, *The Little LISPer*, MIT Press, 1987, Trade Edition. (a good book on recursion)
- Daniel P. Friedman and Matthias Felleisen, *The Little Schemer*, MIT Press, 1995, Fourth Edition. (a good book on recursion)
- Nicolai M. Josuttis, *The C++ Standard Library: A Tutorial and Reference*, Addison Wesley, 1999. (a good STL reference book)
- Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language*, Prentice Hall, 1989, Second Edition. (*the* reference for C, written by its creators)
- Andrew Koenig and Barbara Moo, *Ruminations on C++*, Addison Wesley, 1997. (a great but advanced book on C++ programming)
- Stanley B. Lippman and Josée Lajoie, *C++ Primer*, Addison Wesley, 1998. (a lengthy book introducing C++ language features)
- Stanley B. Lippman, *Essential C++*, Addison Wesley, 2000. (a very short, not-yet-published book covering C++ essentials)
- P. J. Plauger, Alexander A. Stepanov, and Meng Lee, *Standard Template Library: A Definitive Approach to C++ Programming*, Prentice Hall, 1996. (I have not seen this book, but one of the authors wrote the STL.)
- Bjarne Stroustrup, *The C++ Programming Language*, Addison Wesley, 1997. (The creator of the C++ programming language introduces its features.)

4 Assignments and Grades

4.1 Homeworks

Homework assignments give an opportunity both to apply material covered in lecture and to learn new material. Learning the course's material and solving the homework problems are highly correlated.

Most homework problems will be programming problems almost certainly using the C++ programming language. You are free to work on any computer using any compiler or interpreter you desire. Some compilers do not fully support templates or the Standard Template Library; do not use them.

Most homework assignments and solutions will be distributed via the class WWW site. Solutions may be removed after being posted for three weeks; save a copy of you want one. Each assignment will be due at the beginning of the period on the day assigned. Most homework assignments will be submitted electronically, and many homework submissions will be graded by testing your programs' behavior on various inputs. Also, your programs' code may be randomly sampled for correctness and style.

4.2 Class Project

Together, students in CS1321-1 and CS1321-2 will design, build, test, and document, a computer program determining the sales tax for any given group of items sold at a particular location. As far as I am aware, no such program exists, but there is a significant business need for one. If we are successful, we will publish our results.

You are required to participate in at least one aspect of the project. Most tasks will last two weeks. Many tasks will require working with a group of students. An outline of the project and possible tasks will be presented in the second lecture. Sign-ups for tasks will occur soon thereafter on a first-come, first-served basis.

4.3 Quizzes

The purpose of quizzes is to ensure students are not deferring learning the material. As a student in this course, you are academically mature enough that quizzes are not necessary.

4.4 Examinations

Exams are comprehensive but will emphasize the most recent material. Two in-class exams will occur 15 February and 21 March. For 1321-01, the final exam will be given 08 May, 8:30–11:30am. For 1321-02, the exam will be 11 May, 2:00–5:00pm. Please notify me on or before 25 Jan if you have an exam scheduled at the same time.

4.5 Late Work

There are too many students taking the course to accept late work. Late work will not be accepted without a note from a medical doctor or prior notice of an excused absence for a university-sponsored activity.

4.6 Grades

The grades in this course will be determined by the results of

weighting	assignment
45%	exams,
45%	homework assignments, and
10%	class project participation.

Also, the two lowest homework grades will be replaced by grades corresponding to your class project participation grade *even if* your two lowest homework grades are higher. The two reasons for this are: 1) I expect most class project tasks to require about two weeks of work, and 2) this will permit not submitting homework assignments for two weeks if necessary.

Letter grades will be assigned according to the following scale:

90–100	A
80–89	B
70–79	C
60–69	D

with plus and minus grades assigned in marginal cases. Keeping track of your scores will indicate where you stand in the class at any time.

4.7 Collaboration

Many people learn more effectively when they study in small groups and cooperate in various other ways on homework. This can be particularly true in programming assignments, where working with a partner often helps to avoid careless errors. I am very much in favor of this kind of cooperation, so long as all participants actively involve themselves in all aspects of the work—not just split up the assignment and each do only a fraction. When you hand in a paper with your name on it, I assume that you are certifying that this is your work and that you were involved in all aspects of it. Similarly, on your homework paper, you should write the names of the other students with whom you worked.

Here is an example scenario of how a good collaboration might work:

> Both (all) of you sit down with pencil and paper and together plan how you're going to solve things. You go together to a cluster and sit at adjacent machines. When one of you has a problem, the others look over your shoulder. You check after each problem to make sure that the others are all caught up. In your submission, each of you lists the names of all of your collaborators.

If you're worried you are becoming too dependent on your partners, do the planning together, but go sit at a machine that's not nearby. See what you can do. In general, I strongly encourage you to work as a group. It's a very effective way of catching conceptual and other errors and of refining one's thinking and understanding.

Unless otherwise specified, each student must submit her/his own homework solution. Collaboration such as that listed above, e.g., planning the programming project together is permitted, but each student must submit her/his own programs. The names of all students with whom you worked must be listed.

(The preceding is based on the MIT 6.001 collaboration policy.)

5 Academic Violations

The first and all subsequent violations of academic integrity will immediately result in a grade of F for the course.

Among the possible violations are cheating, counterfeit work (submitting work created or produced by others including submitting joint work as one's own), and plagiarism. Not listing the name of a collaborator will be deemed cheating. Similarly, copying another person's work and representing it as one's own work is a violation.

There has been much recent work on automatic detection of plagiarism. Your assignments may be checked using these tools. Among these are plagiarism.org and Glatt Plagiarism Services. Detection of plagiarism by these tools will be considered sufficient evidence of a violation of academic integrity.

You are also expected to abide by the Trinity Code of Ethics for Computing.

If you cannot abide by these rules, do not take this course from me.

5.1 Security of Your Work

You are responsible for ensuring the security of your work. This includes both physical and electronic copies. Even though the Trinity Code of Ethics for Computing prohibits unauthorized access to your computer files, you are responsible for ensuring they are adequately protected. For information how to restrict access to your Trinity CS computer files, see File Security Tutorial.

6 Disabilities

I encourage students with disabilities including "invisible" disabilities like chronic diseases, learning disabilities, and psychiatric disabilities to discuss with me appropriate accommodations that might be helpful.