

# CSCI0320 Student Missive

Introduction to Software Engineering  
CSCI0320, Summer 2009  
Steven P. Reiss  
John Jannotti

## 1 Purpose

The purpose of this course is to provide students with the background they need to design and implement moderate (10-100k loc) software systems. The emphasis of the course is on learning the design and coding techniques that are needed in the implementation of such systems. Design techniques include understanding design from the class level through to software architectures, understanding how to express designs, understanding how to design user interfaces as part of such systems, understanding how to use threads and sockets, understanding file I/O and databases, and understanding how to integrate applications with the web. Coding techniques include proper use of tools, an emphasis on testing, memory management, systems programming, proper use of programming languages, and coding and programming styles. The course also serves as a basic introduction to software engineering, with students doing a team project from requirements through implementation. Here emphasis is on learning how to design for and work as a team, doing presentations, and that the primary goal when writing code is to enable it to be maintained.

## 2 Who Should be Taking the Course

Students who are interested in software design and engineering principles should take this course as an introduction. It is excellent preparation for a summer internship or future computer-related job, as it emphasizes skills required to work well in a team. Students outside the department take this course to develop the skills necessary for participation in a group project involving a large amount of programming.

We expect the students taking this class to have a strong sense of OO programming, working knowledge of UNIX, good understanding of data structures, and to be highly motivated. If you have taken CS 15/16, 17/18, or 19, or equivalent you should be sufficiently prepared for this course (CS22 and CS31 or equivalent will help as well). If you have not taken the prerequisites, you should talk to one of the professors to determine if the course is right for you.

## 3 Content

The course basically covers four topics: advanced programming concepts, object-oriented (OO) design, software engineering, and systems programming. These components will be blended throughout the course, with the emphasis throughout being on understanding program and system design at all levels. The beginning of the course will emphasize programming concepts and small-scale design issues. The middle of the course will cover advanced programming using threads and sockets as well as system design concepts. The latter part of the course will cover systems programming issues, tools, and concepts.

## 4 Lecture, Labs, Presentations, and Sections

In this course we will use a variety of different approaches to teaching design and programming. We will have traditional lectures, given by Professors Reiss and Jannotti. These will be focused on design issues and should provide the basic principles and techniques you will need to master in order to design the latter programming assignments and especially your final project. Attendance in class is mandatory.

Lectures, however, aren't the best way of teaching hands-on material. Programming techniques and concepts are best learned through doing. Rather than having massive programming assignments that cover all the topics that are necessary, we will have lab homeworks. These consist of a sequence of instructions and small tasks that should help you learn how to use some of the basic tools and practice advanced programming techniques. Because of time considerations, these will be done as homeworks.

An essential part of doing a large project and of software engineering in general involves doing presentations, telling others what you propose to do, what you are doing, and what you have done. We will provide you with the opportunity to make such presentations throughout the course, concentrated primarily on your final project.

## 5 Textbooks

The textbook should be available in the bookstore.

Vermeulen, et al, *The Elements of Java Style*

This text, which is required, provides a set of style and coding guidelines that is a good starting point for serious Java programming. (We will provide our own additions and suggestions in handouts and on the web as the course progresses.) While this book is restricted to Java, most of the principles apply to C and C++ programming as well.

Some other texts that might be of interest include:

Whittaker, *How to Break Software*

This book provides a set of guidelines for testing software, which is a topic we will stress throughout the course. Thinking about testing as done in the book helps you produce more robust software with fewer errors that requires less debugging time.

Braude, *Software Design*

This text provides a nice compact overview of several topics that we will cover in the course in parts I and II. These include a nice overview of software engineering, various design issues, UML, and design patterns.

## 6 Getting Help

Right now we do not have a TA for the course, although that might change. If you need help on an assignment, lab, or your final project, you should contact one of the professors. In addition, we will allocate time during class (probably the first 30 minutes) for asking questions. Professor Reiss is generally available in his office (CIT 403) from 8-4 on weekdays. Professor Jannotti is available in his office (CIT 379) generally from 10-6.

## 7 Required Coursework

This course is comprised of labs, programs with graded designs, and a final project.

The labs are designed to teach important course concepts. Labs are mandatory. They shouldn't take more than an hour or two to complete.

The programs will range from smaller individual projects, meant to familiarize you with advanced Java programming and object-oriented design techniques, to larger collaborative projects. They will cover advanced programming, use of available programming tools, and user interface and web programming, and will culminate in a substantial group project at the end of the semester.

**In order to pass CSCI0320, you need to have a passing final grade in the course, and you also need to hand in a working version of all assignments.** Failure to hand in a working version of all programming assignments (regardless of your final average) will result in an automatic NC. The last day to hand in a working version of an assignment is Thursday, August 6th at 4 pm. See Unacceptable handins below for more information.

Programming assignments are not merely graded on their functionality: there will be a special emphasis on good software engineering practices. This means that we will expect you to produce readable, well-documented code, to use modularity and data protection, and to spend time thinking about the design and breakdown of your program before you start coding. Your code will be read! A perfectly working program which is not well-designed, well-documented or readable will not receive an A.

**Extra Credit:** There will be ample room for bells, whistles, and other credit-garnering efforts on the part of ambitious programmers. You are invited to get creative, as long as it does not make you late. Remember to always implement the program according to assignment specifications before launching off into extra credit work. Rewarding bells and whistles with extra credit is left to the discretion of the professors, so you may want to discuss ambitious plans beforehand to make sure they are considered appropriate for credit. In general, efforts outside the intent and focus of the assignments will receive small amounts of credit.

**Handing In:** Your working code should be handed in online by the deadline announced in the assignment's handout; the exact handin procedure on each handout will be announced at the time. You must hand in your assignment from your account.

**Unacceptable handins:** Occasionally, a student will hand in a program that is so far from meeting the requirements of the assignment that it will be considered unacceptable. In that event, the assignment will be registered as not done. In order to pass CSCI0320, you need a passing final average and you need to have handed in a working/acceptable version of all the assignments. Even if you have a passing average, you will still receive an NC for the course if you do not hand in an acceptable version of all the assignments by Thursday, August 6 at 4pm. This deadline is set by the university and will be strictly adhered to without exception. Since it is extremely hard to state explicitly what is and is not considered acceptable, it is the student's responsibility, when his or her work is not at all close to meeting the assignment's specifications, to check whether the work meets minimal requirements by conferring with one of the professors. You can keep in mind that any program with no functionality or any program with poor coding style (i.e. no comments and in severe violation of the CSCI0320 coding standards) will automatically be considered 'unacceptable.'

## 8 Grade Breakdown

The course grade will be based on the in-class projects, programming assignments, class participation and the final project. The following percentage breakdown will be used:

Program 1 (Galaxy)	15%
Program 2 (NewsWhere)	15%
Program 3 (FileView)	15%
Final Project	30%
Labs/Homework	20%
Class Participation	5%

Designs will be counted as a percentage of that project's grade.

**Group Grading Policy:** We far prefer that each group has a single electronic handin per program. The grade assigned will be a group grade, i.e. the same for all members. However, occasionally the situation arises where one team member feels that another member's contribution is substantially deficient. In such cases, the team members should first make a serious effort to deal with this problem and to hand in a working program together; indeed, part of working in a group is resolving differences between group members, and motivating your teammate(s) to work in an expeditious manner. Failing this, the person whose work is completed may submit a separate handin. The README should indicate that you are handing in separately, and the professors should be notified via email. Next, this person must meet with the professor and demonstrate that his or her portion of the program works. This will probably necessitate writing some test code. Finally, at some point the group should hand in the integrated code for the program. This should be handed in by the other person in the group to avoid overwriting the first partial handin.

If you believe that you are in this situation, you are strongly encouraged to discuss it beforehand with one of the professors. In addition, we reserve the right not to accept separate handins from a group if any of the above conditions are not met.

## 9 Deadlines and Late Handins

Assignment deadlines are intentionally spread out because the course is intensive. You will find that if you start each assignment when it is handed out (rather than near when it is due), the workload will be manageable. However, if you are late with one assignment and that causes you to compress the work on the next, you will probably find that the course load will quickly become unbearable. Moreover, the latter assignments are team efforts, and to be fair to your other team members, you must not get behind in the course material.

**Deadlines:** Tentative due dates for the assignments are listed in the syllabus, but always note the actual due date on an assignment handout when it goes out and note any changes announced in class.

**Late Days:** We will give you two "late days" to use at your discretion over the course of the semester. Each provides you with a 24 hour extension on a programming assignment. If you use late days on a group project, they apply only to you and not to the group as a whole.

**Late Handins:** Late handins, while discouraged, are allowed. However, your grade will then be subject to penalty, depending on the lateness of your handin. A percentage of the score you receive will be deducted as follows:

HANDIN BY	PENALTY
0-24 hours late	10%
24-48 hours late	20%
48-72 hours late	30%

72-96 hours late	40%
after 96 hours	100%

There are several remarks on the late policy. First, this policy does not apply to the final project, homeworks, labs, or for designs, for which there are no late handin possibilities. Second, late penalties have nothing to do with the determination of an assignment as working/acceptable or unacceptable. And finally, please note that handins later than four days after the original deadline will not receive any credit, and will thus endanger your chances of passing the course. It has been shown in the past that CSCI0320 is simply too tight a course for people to be able to fall behind and still do well. While this late policy may seem harsh, it is designed to make you recognize the importance of meeting your deadlines.

There is no extra credit for handing in early.

**Extensions:** As a general rule, no extensions will be granted in CSCI0320. Academic overload is *not* a valid excuse for an extension. Extensions *may* be granted by the professor only for medical or other reasons which are deemed acceptable. In the case of medical excuses, a note from Health Services is required; in other cases, a note from a dean would be appropriate. Of course, extensions for individual members on team projects are a more complicated issue, and these will hardly ever be given. To request an extension, talk to one of the professors.

**Incompletes:** Incompletes for the course will be granted only in extreme circumstances. You should be aware of the university policy that says that students can only be granted an Incomplete if their work up until the date of the Incomplete is satisfactory. If you need to file for an incomplete you should talk to one of the professors.

## 10 Accounts

All students will fill out a registration form on the first day of class, so we have your name recorded, and whether or not you require a new CS account.

## 11 Staying Up to Date

You are required to attend each class and are responsible for all announcements, etc. made in class. We will attempt to establish an electronic bulletin board, probably a mailing list, that will also be used for communication. You are then responsible for monitoring this. We will try to keep the web site up-to-date as well.

## 12 Getting Your Questions Answered

There are many ways to get questions answered or to get problems solved. For the most part you are encouraged to become self-sufficient and learn to use the documentation and resources that are available. This is typically how programmers work and learning to use these resources effectively will make you a better programmer. To help you we note the following:

**CSCI0320 Documentation Page:** We have created a page off of the CSCI0320 home page which contains links to many useful resources and sources of documentation. Also, if you find any good sources of online

docs, let us know, and we can add that link off the docs page.

**JavaDoc:** JavaDoc for all the standard Java packages is available at <http://java.sun.com/javase/6/docs/api/>. JavaDoc locations for the various support libraries for the assignments will be listed in the assignment hand-outs or on the website.

**CSCI0320 FAQ Page:** If you have a question, problem, or something to say which you feel may be of interest to other students in the class, post it to the CSCI0320 mailing list. The professors will regularly read and even respond to these postings. Note that it is in your best interest to frequently read the postings in the CSCI0320 FAQ, and many will be pertinent to your assignments.

Questions about packages used for assignments can be asked in the mailing list *after* you make an attempt to find an answer in the documentation, either online or available on paper in the lab. Feel free to answer questions asked by other students or point them to helpful documentation you've found, but be careful not to share information that's not permitted by the collaboration policy. You should not use the mailing lists to share code, ideas, or algorithms, nor should you make non-course related postings to the mailing list.

**E-mail:** If you have a personal concern or a question which isn't of interest to the class at large, you can mail the staff about it by sending mail to [cs032tas@cs.brown.edu](mailto:cs032tas@cs.brown.edu). Professor Reiss's email is [spr@cs.brown.edu](mailto:spr@cs.brown.edu); Professor Jannotti's is [jj@cs.brown.edu](mailto:jj@cs.brown.edu).

## 13 Working from Home

We will make an effort to allow you to work on your own machines for many of the projects. This is easier on the Java assignments than on the C/C++ ones which are more OS-dependent. Note however, that we cannot be responsible for handling the complete set of different environments and configurations that are possible. We will provide libraries and some configuration help, but you are expected to be able to do much of this on your own.

We will also make machines in the CS department available as needed.

## 14 Feedback

We're always interested in ways to make CSCI0320 a better course. You are urged to bring any suggestions and/or comments you may have regarding the course, the assignments, the lectures, the labs, the practicums, or anything else to one of the professors. All comments (both positive and negative) will be much appreciated.