LEARNING ABROAD CENTER
UNIVERSITY OF MINNESOTA

In Partnership With

FOM
FUNDACIÓN JOSÉ ORTEGA Y GASSET - GREGORIO MARAÑÓN

Program Design & Development

Course Details

Course Designator & Number: MADR 3081W

Number of Credits: 4

Language of Instruction: English

Contact Hours: 60

Instructor: Onsite Faculty

Course Description

Principles of programming design/analysis. Concepts in software development. Uses a programming project to illustrate key ideas in program design/development, data structures, debugging, files, I/O, testing, and coding standards.

This course covers skills, tools, and theory related to becoming a good software developer. The course will prepare students to succeed in 4xxx- and 5xxx-level programming intensive courses, especially those that are programming-intensive and/or use C/C++. This is a required course for CSci majors and a “project” course for non-CSci majors.

Course Objectives

Upon successful completion of the course, students should have experience with the following:

1. The software engineering process, for example, a general understanding of large programming projects and their differences from small ones, as well as different software process models, design basics, and testing basics.
2. Reasonable C++ programmer proficiency. In particular, this means being able to use C+class basics, inheritance, polymorphism, assertions and exception handling, pointers, memory management, and similar advanced topics.
3. UML class diagrams and other fundamental diagrams.
4. A large, mult-iteration programming project.
5. A few software development tools such as version control software, debuggers, etc.

6. Good coding practices such as documentation skills, good program organization, use of naming conventions, good class design, etc.
7. Writing in computer science. This writing could take a variety of forms and might involve items such as project descriptions, code documentation, and progress reports.

Methodology

CSci 3081W has a number of goals: It is an advanced programming course, it involves substantial program design, it uses the C++ programming language, it involves professional communication, and it provides an introduction to software processes and tools, etc. In particular, a major goal of this class is to provide a first academic experience with "programming in the large." Specifically, the large programming project in 3081W will involve most if not all of the following: substantial design, possibly working in a team with other programmers, multiple iterations, many files of software code rather than one or two, linking with external libraries, code documentation. and use of good programming standards and practices.

Course Prerequisites

The prerequisites for 3081W are CSci 2021 (Machine Organization), CSci 2041 (Advanced Program Principles), and CS upper division standing. Students should have learned some C as well as the basics of pointers and memory management. 3081W will build on this in its coverage of C++ and more advanced memory management techniques. From 2041, students should have a general knowledge of programming techniques and languages, which will be helpful when doing the programming in 3081W.

CSci 3081W is a prerequisite for CSci 4271W, 5123, and 5607. However, even though it is not a formal prerequisite, the department suggests taking 3081W before or simultaneously with 4061, since the C/C++ programming in 3081W might be useful in 4061. Moreover, the department suggests taking 3081W before most electives since the programming skills learned in 3081W will likely be useful in elective classes that involve significant amounts of programming.

Here are the classes for which 3081W is a formal prerequisite:

- CSci 4271W, Development of Secure Software Systems. The C/C++, advanced programming, and writing skills learned in 3081W will be useful in 4271W.
- CSci 5123, Recommender Systems. The programming skills learned in 3081W, especially those that involve working with large systems, will be useful in 5123.
- CSci 5607, Fundamentals of Computer Graphics I. The C/C++ and advanced programming skills learned in 3081W will be useful in 5607.

Students should take CSci 3081W before taking most electives. For most students this means taking 3081W in their junior, not their senior, year. CSci 3081W is restricted to CS majors.

Other students who wish to take 3081W for some valid reason should contact a department adviser, not the course teacher.

Required Reading / Materials

Class Format

CSci 3081W is a four-credit course, with three hours/week in lecture, plus a one-hour lab once per week. Labs are usually used for students to work on C++ practice problems, or to work on larger programming projects in a structured environment.

Possible Text(s)

- *Code Complete*, by Steve McConnell, published by Microsoft Press.
- *UML Distilled: A Brief Guide to the Standard Object Modeling Language,* by Martin Fowler, published by Addison-Wesley.

Communication Skills

Communication skills play an important part of large program design and development, and so they play an important role in the course. CSci 3081W is a writing-intensive class, and must therefore fulfill the writing intensive criteria: writing must play an integral role in the course, a good amount of the coursework must involve some type of writing (although writing can be widely interpreted to cover items such as UML diagrams and code comments), students should not be able to pass the course without meeting a minimal standard of writing proficiency on the coursework, and the course must involve some relevant writing instruction.

The writing work in 3081W can take a number of forms, including design documents, high-level system descriptions, project progress reports, code summaries and documentation, and short answer problem solutions. Teachers are free to choose whichever types of writing fit well into the course.

Writing Intensive Guidelines

Design and analysis of computer programs concerns all aspects of software development ranging from capturing the business case and technical rationale to detailed documentation of what the software does, how it shall be used, and how it is constructed. Excellent writing skills are a necessity to make larger software development projects successful. In this software-engineering course, the students must learn the critical skills of how to adequately specify and document software projects and software artifacts.

Assignments for this course will be reviewed for technical content, clarity, and quality of writing. These include, but are not limited to:

- Requirements Document: This document provides the rationale for, goals with, and specific requirements on a new software system. This is a written document where clarity is imperative.

- Users Manual: The users manual for software systems is often a very weak point. This document will capture how the software system will be used in an easy to understand and clear form.

Draft versions of these documents are required and will be critiqued by the instructor.

Grading

Grading Rubric

Letter Grade	Score or Percentage	Description
A	93–100	Achievement that is outstanding relative to the level necessary to meet course requirements.
A-	90–92	Achievement that is significantly above the level necessary to meet course requirements.
B+	87–89	
B	83–86	
B-	80–82	Achievement that meets the course requirements in every respect.
C+	77–79	
C	73–76	
C-	70–72	Achievement that is worthy of credit even though it fails to fully meet the course requirements.
D+	67–69	
D	60–66	
F	0–59	Represents failure (or no credit) and signifies that the work was either (1) completed but at a level of achievement that is not worthy of credit or (2) was not completed and there was no agreement between the instructor and the student that the student would be awarded an I.

Project Grading

A large part of the grade (tentatively 40%) will be based on the project in this course. The content, format, and clarity of the various documents produced as part of this project determines the project grade. Students have an opportunity to receive feedback and later revise their work.

User Manual Project

In this course, students will create a User Manual. This document should be a self-contained description of how to use a system, and should be a polished, professional piece of technical prose that a software company is proud to have accompany one of their products.

The document should have a structure that is evident to someone who is reading it straight through and looking for a particular topic or fact. A table of contents is required, and the organization that it reflects should be considered carefully. An Index and Appendices might also be helpful.

The document should be completely self-explanatory. Do not assume the readers have functional specifications. Students' previous documents may be edited and revised when creating this User Manual.

The following list of topics must be covered, but the order and style in which these topics are presented may be left to the students' discretion:

- **Introduction:** a concise statement of what the program does, possibly including motivation and philosophy.
- **How to use your system:** an overall description of the style of user interaction, device constraints, and any arbitrary design choices made that the user ought to know about.
- **Detailed system operation:** an organized list of all user commands and when they are appropriate; some examples might be helpful. A section for novices and experts are also possibilities.
- **Error recognition and handling:** what to expect and what to do when it happens.
- **A list of known features [actions that some would call bugs!] and deficiencies.**

Required Lectures and Presentations

There will be two general lectures on technical writing and technical presentations in this course.

Course Content

Unit 1

- Course Introduction, C++ Basics

Unit 2

- C++ Classes, UML Diagrams, Professional Writing in CS

Unit 3

- Programs and Class Design, Makefiles, Pointers and Polymorphism in C++

Unit 4

- Project Iteration 1, More Pointers, Code Style Guidelines

Unit 5

- Constructors and Destructors, Parameter Passing, Introduction to Design Patterns

Unit 6

- Miscellaneous Topics, Project Issues

Unit 7

- Software Development Methodologies, Project Iteration 2

Unit 8

- Copy Constructors and Overloaded Assignment Operators, Code Style, Midterm

Unit 9

- Code Style, More Design Patterns

Unit 10

- Project Reports, Testing Strategies, Debugging

Unit 11

- C++ Templates and the Standard Template Library, Project Iteration 3

Unit 12

- More Design Patterns, Project Documentation, Refactoring

Unit 13

- More Testing

Unit 14

- Advanced UML, More About Design, Class Retrospective

Policies

Attendance Policy

Students are expected to be on time and attend all classes while abroad. Many instructors assess both attendance and participation when assigning a final course grade. Attendance alone does not guarantee a positive participation grade; the student should be prepared for class and engage in class discussion. See the on-site syllabus for specific class requirements.

University of Minnesota Policies & Procedures

Academic integrity is essential to a positive teaching and learning environment. All students enrolled in University courses are expected to complete coursework responsibilities with fairness and honesty. Failure to do so by seeking unfair advantage over others or misrepresenting someone else's work as your own can result in disciplinary action. The University Student Conduct Code defines scholastic dishonesty as follows:

Scholastic Dishonesty

Scholastic dishonesty means plagiarizing; cheating on assignments or examinations; engaging in unauthorized collaboration on academic work; taking, acquiring, or using test materials without faculty permission; submitting false or incomplete records of academic achievement; acting alone or in cooperation with another to falsify records or to obtain dishonestly grades, honors, awards, or professional endorsement; altering forging, or misusing a University academic record; or fabricating or falsifying data, research procedures, or data analysis.

Within this course, a student responsible for scholastic dishonesty can be assigned a penalty up to and including an “F” or “N” for the course. If you have any questions regarding the expectations for a specific assignment or exam, ask.

Student Conduct

The University of Minnesota has specific policies concerning student conduct. This information can be found on the Learning Abroad Center website.