

C-S 421: Programming Language Concepts

Dr. Kenny Hunt

Office: Wing 213

Phone: 785.6822

WWW: charity.cs.uwlax.edu

Email: hunt.kenn@uwlax.edu

Office Hours: See above web site

Textbook: Tucker/Noonan, *Programming Languages, Principles and Paradigms*

Course Objective

The aim of this course is to provide students with a basic understanding of programming language design. The material will emphasize the conceptual *design* and *implementation* of programming languages. We will survey languages representative of the major paradigms (logical, functional, imperative) in moderate detail.

How C-S 421 Serves Departmental Objectives

The educational objective of the CS Department is that graduating students be able to analyze problems, design software architectures, implement the design in the form of a program and communicate all relevant technical information both verbally and in written form. This course directly serves these educational ends by introducing implementation choices not necessarily covered in other courses and by providing detailed coverage of language implementation.

Requirements and Grading

Course grades will be based on approximately 6 homework assignments, two "mid-term" exams, and a final exam. Grades will be computed using the weighting scheme listed below. Homework assignments will typically consist of both a programming component and a "written" component. ***Any student receiving a score of less than 50% on any three assignments will fail the course regardless of their performance on any other portion of the class.***

Description	Percentage
Homework Assignments (approximately 6)	30%
Exam 1	20%
Exam 2	20%
Final Exam	30%

All homework assignments are due **before class** on the due date (unless otherwise specified). Assignments turned in within 24 hours after this time will receive a **50% penalty** and those turned in more than 24 hours late will receive **no credit**. Exceptions may be granted under extraordinary circumstances if a student requests an extension **prior to** the deadline. If a request cannot be made prior to the deadline due to a medical (or possibly other type of) emergency, the student must request an extension as soon as possible. Students should not assume that exceptions to the grading policy have been or will be given *simply because they have been requested*. Exceptions to the grading policy will be made exclusively and explicitly via email or other *written* communication.

If you wish to receive an acknowledgement that I have received your email, place the word "acknowledge" in the subject line. *Make sure that it is spelled correctly.*

The following table provides a rough description of the grading policy for *programming* components of homework. All code must be tested, documented and submitted via email.

Programming Assignment Grading Policy	Score
A working program that is submitted via email. A test document (list of limitations and test results) is also submitted.	100
Same as above but missing a test document	90
A program that compiles but fails to meet minor requirements	80
A program that compiles but fails to meet major requirements	60
Maximum score for a program that is submitted one day late	50
Nothing submitted or a program is submitted more than 24 hours late or the program doesn't compile	0

The following table provides a rough description of the grading policy for *written* components of homework. Written components do not need to be emailed.

Description	Score
A neatly formatted text with correct answers to assigned questions	100
Maximum score for an assignment submitted one day late	50
Nothing submitted or the assignment is submitted more that 24 hours late or the assignment is sloppily written	0

Expectations

Of the Teacher

- Students can expect that I will *start* and *complete* class lectures according to the published schedule and will be well prepared to deliver all class lectures.
- Students can expect that I will be respectful at all times.
- Students can expect that I will be prompt, fair and approachable when assessing student *performance*¹.
- Students can also expect that I will be present during published office hours and generally available outside of office hours if other arrangements are requested.

Of the Students

- I expect each student to be present at the beginning of each scheduled class time. Although attendance is not graded, perfect class attendance is expected of all students.
- I expect students to be well prepared for each class lecture by having completed any assignments and read relevant portions of the text under study.
- I expect students to be respectful of others during class time by being attentive and refraining from distracting personal conversations.
- I expect students to be responsible for all material presented in class as well as all material covered in the reading assignments when taking an exam or quiz.

¹ Please distinguish assessments of *performance* from assessments of *effort* which neither I nor anyone else can adequately judge

- I expect students to complete and submit assigned work by the required date.

In addition, programming courses are notoriously more time consuming than most. You should expect the programming assignments to take a significant amount of time to complete. Although the time and effort spent on projects will vary from student to student you should expect to spend an **average** of ten hours per week on programming assignments. For instance, if you are given two weeks to complete an assignment it is expected that the assignment will take the average student approximately 20 hours to complete.

Course Schedule

Please note that the schedule below is only approximate.

Date	Description	Reading Due
Week 1	Introduction	Chapter 1
Week 2	Syntax	Chapter 2
Week 3	Syntax and Semantic Definition	Chapters 2 and 3
Week 4	Types and Semantic Definition	Chapters 2 and 3
Week 8	Imperative Programming (C/C++/Fortran)	Chapter 4
Week 9	Memory Management	Chapter 5
Week X	Exception Handling	Chapter 6
Week 10	OOP (C++/Java/Smalltalk)	Chapter 7
Week 5	Functional Programming (LISP/Scheme/ML)	Chapter 8
Week 6	Functional Programming (LISP/Scheme/ML)	Chapter 8
Week 7	Functional Programming (LISP/Scheme/ML)	Chapter 8
Week 11	Logic Programming (Prolog/Haskell)	Chapter 9
Week 12	Logic Programming (Prolog/Haskell)	Chapter 9
Week 13	Concurrent Programming	Chapter 9
Week 14	Concurrent Programming	Chapter 9

Academic Integrity

Homework and programming assignments, except when explicitly specified otherwise, should be done alone.

It is reasonable to discuss general approaches to problem solutions or algorithm design with other students but the bulk of the work must be done alone. Working out details, sharing in the write-up or sharing or copying code will be treated as a violation of the academic integrity rules. *I will aggressively pursue appropriate disciplinary action for any form of unethical behavior.* The UWL procedures for handling academic misconduct can be obtained from the [Office of Student Life](#).

Examples of ethical collaboration:

- Providing help with operating system questions such as how to name a file, how to create and manipulate folders, how to run a program, or how to print files.
- Assisting in the detection of syntax errors.
- Helping to determine why a running program crashes or fails, most likely by tracking down the point at which an error occurs.

Examples of unethical collaboration

- Sharing source code or other homework solutions.
- Giving specific methods or algorithms for solving a particular problem.
- Copying programs from news groups or other network resources.

Any student with a documented disability (e.g., physical, learning, psychiatric, vision, or hearing, etc.) who needs to arrange reasonable accommodations must contact the instructor and the Disability Resource Services Office (165 Murphy Library) at the beginning of the semester. Students who are currently using Disability Resource Services will have a copy of a contract that verifies they are qualified students with disabilities who have documentation on file in the Disability Resource Service Office. This material is available in alternative formats upon request.