CIS-280: Program Development Techniques

Instructor: Sandra Carberry
Office: 456 Smith
Office Hours: 12:45-1:45pm Tuesday
2:00-3:00pm Thursday

Teaching Assistant: Michael Smith
Office: 115B Pearson Hall
Office Hours: Monday 12:15-1:15pm
Wednesday 11:45am-1:45pm

Teaching Assistant: Manabu Torii
Office: 115B Pearson Hall
Office Hours: Friday 10am-noon
Monday after last lab (in lab room)

Web page: www.cis.udel.edu/∼carberry/CIS-280
Project Number: 2081

1 Course Description

CIS-280 is a rigorous introduction to the methodology and paradigms of computer science. Although the course will involve a great deal of programming to exemplify and gain experience with the concepts being taught, the emphasis of the course is not on a particular programming language. Instead, the course will focus on programming paradigms and concepts, including data and procedural abstraction, modularity, object-oriented programming, functional programming, data-directed programming, and algorithm complexity.

Mastery of the concepts in this course is essential for good software design, since a software designer must be able to consider different design options and select those that are most appropriate for the problem at hand. Thus this course provides the foundations for addressing complex programming projects in subsequent courses, particularly in the capstone software engineering course required of all students pursuing a Bachelor of Science degree in computer science.

Why Scheme?

Programming assignments will be done in Scheme, a dialect of Lisp. Scheme is a wonderful language in which to illustrate and experiment with the programming paradigms introduced in CIS-280. First, the syntax of Scheme is easy to learn, so little effort needs to be devoted to teaching the language itself. Second, Scheme removes the programmer from many of the details involved in setting up and using data structures in most conventional programming languages such as C++ — thus one can focus on the concepts and paradigms being studied. And most important, Scheme supports the modular decomposition strategies and incremental design techniques that we will be studying.

Objectives

- Have a command of the major techniques for designing programs and controlling complexity
- Be able to use modern approaches to dealing with time computationally (objects with state, functional programming, concurrent programming, lazy evaluation, nondeterministic programming)

- Understand the issues associated with programming language design and implementation
- Have a mastery of a non-standard programming language

Prerequisites

CIS-220: data structures
Competency with Unix

Textbooks

Abelson, H. and G. Sussman with J. Sussman, *Structure and Interpretation of Computer Programs* (second edition)

2 Grading

Exam I	15%
Exam II	20%
Final Exam	20%
Homework Assignments	30%
Project	15%
Labs	See below for lab policy

There will be two midterm exams and a final exam. Students enrolled in sections 010 and 011 will be required to remain at the midterm exams for the entire period.

Lab Work

Labs meet on Monday in 009 Willard Hall. During labs, you will be working on exercises that illustrate the concepts discussed in class the preceding week; these exercises will help you with the homework assignments. At the end of lab, you should submit your work on the course submission web site unless the assignment specifies a different method of submission. If you attend lab and work conscientiously on the lab assignment, then you will be given credit for that lab even if you do not finish it; if you miss lab, you must correctly complete the lab assignment in its entirety and submit it by the start of lab the following Monday in order to be given credit for that lab. **(Late labs will not be accepted.)** One point will be deducted from your final grade for each lab for which you do not receive credit (either by attendance at the lab, or by correctly completing the lab assignment and submitting it by the start of lab the following Monday). **A few labs will not have associated lab assignments and thus you must attend those labs in order t**

Homework and Project

Homework assignments are intended to give you an opportunity to work with the concepts discussed in class. Assignments will be distributed throughout the semester. The project is a longer programming assignment, requiring substantial effort. Unless the assignment or the course web page says otherwise, all assignments must be submitted through the course submission web page.

Both homework assignments and the project are due **prior to the start of class** on the announced due date. They may be submitted without penalty **prior to the start of the next class**.

Assignments submitted after the no penalty extension will be penalized as follows:

TURNED IN	PENALTY
9am, one day after the no penalty extension	25%
9am, two days after the no penalty extension	50%
9am, three days after the no penalty extension	75%
after 9am, three days after the no penalty extension	not accepted

Saturday and Sunday will not be counted in determining the number of late days. Thus an assignment due on Tuesday but **SUBMITTED** prior to the start of class on the following Thursday will not be penalized, an assignment due on Tuesday but **SUBMITTED** by 9am the following Friday will be penalized 25% of the total points that the assignment is worth, and an assignment due on Tuesday but **SUBMITTED** by 9am the following Monday will be penalized 50% of the total points that the assignment is worth. Please note that penalties are based on when the assignment was **SUBMITTED**.

Important Policies

- As noted under "Homework and Project" above, assignments must be submitted prior to the start of class. Once class has begun, the assignment is late and will be penalized as specified in "Homework and Project" above.
- Students must do their work independently. This means that you may not work together with another student on homework assignments or the project, and you may not examine another student's code or homework assignment. You must protect your password, and you may not give your password to another student or leave your assignments so that they are accessible to another student. Both the student who turns in work that is not his or her own and the student who allowed his or her work to be used by another student are guilty of cheating; such cases will be handled according to University policies on academic dishonesty.
- However, students are encouraged to discuss the course material with one another. In addition, you are free to discuss debugging strategies with one another. But assignments must be done independently.

3 Lectures and Readings: Abelson and Sussman

Chapters 1 through 3 will be covered, along with portions of chapters 4 or 5 if time permits. An updated reading list will be distributed as the semester progresses.

DATE	TOPIC	TEXT
Feb. 8	Introduction to CIS-280	
	Basic Concepts	pp.1-13
Feb. 10	Substitution Model, Simple Recursion	pp. 13-31
Feb. 14	LAB: Exercises on Basic Concepts and Recursion	
Feb. 15	Procedures and Processes	pp.31-42
	Complexity and Orders of Growth	pp.42-46
Feb. 17	Algorithms: GCD, Primes	pp. 48-52
Feb. 21	LAB: Exercises on Recursion and Complexity	
Feb. 22	Probabilistic Methods	pp.51-53