CS010 – Introduction to Computer Science I
Fall, 2007

Course Information

OVERVIEW: The course introduces students to the field of Computer Science via *programming*. We will focus on the language *Scheme* without assuming any prior programming experience. Using Scheme programming as our vehicle, we will tour various concepts from CS including: design, logical thinking and problem solving, algorithm development, abstraction, software development process, and the object-oriented paradigm. Although the majority of time will be spent learning and using Scheme, the focus throughout will be on learning foundational concepts of problem-solving and design. From the Westmont catalog:

> "No prior computer or programming experience required. Basics of programming including language features, disciplined programming style, and documentation. Problem solving, algorithm design, and the software development process."

PREREQUISITES: Fulfillment of the mathematics competency requirement.

COURSE OBJECTIVES:

- Learn disciplined design skills
- Be able to reason and problem-solve using abstractions (GE, II.D.)
- Implement solutions to problems as a Scheme program

In order to solve a problem of any significance, you must first understand the problem at a fairly deep level and then you must *design* a series of actions that embody a solution to the problem. In this class, you will learn and exercise design skills. As one of our tools, we will use *abstraction* to make complex problems manageable. This class satisfies the *reasoning abstractly* requirement of the GE. In the context of this class, abstraction takes several forms including functions, meta-functions, modeling, information hiding, hierarchical decomposition, and replication and repetition.

This class also provides an entry point to the major and minor in Computer Science. In alignment with the departmental learning outcomes in computer science, students should: begin to learn to *think creatively and flexibly*; improve their ability to *communicate effectively*; start acquiring the *fundamental concepts and skills* of Computer Science; and identify initial *relationships between Christian faith and Computer Science*. In addition, students should learn to identify the promise and peril of technology and discern appropriate technologies for particular problems (Westmont Learning Standard 6).

The successful student in this course will have the programming skills to take moderately scaled problems, formulate an algorithmic solution, and implement that solution in the Scheme programming language. Such students will also have acquired the conceptual framework for continuing study in Computer Science.

LOCATION: Bauder Hall 101

DAYS & TIME: Tuesday and Thursday, 10:00-11:50 pm.

TEXTS: [**required**] *How To Design Programs: An Introduction to Programming and Computing*, by Matthias Felleisen, Robert Bruce Findler, Matthew Flatt, & Shriram Krishnamurthi. MIT Press: Cambridge, Mass. (2001).

[**optional**] *Hackers and Painters: Big Ideas from the Computer Age*, by Paul Graham. O'Reilly Media, Inc. (2004).

COURSE WEBPAGE: http://www.westmont.edu/~iba/teaching/CS010

Instructor Information

INSTRUCTOR: Dr. Wayne Iba
OFFICE: Mathematics and Computer Science Modular Building
OFFICE HOURS: TBD
EMAIL: iba@westmont.edu
PHONE: (805)565-6799

Course Assignments, Requirements and Policies

EXPECTATIONS: This class requires a greater than average time commitment for many students. If you are unable or unwilling to invest that time, you will probably struggle; please reevaluate your priorities at the beginning of the semester. Before you come to class, I expect you to be familiar with the sections of the text to be covered each day and to be prepared to contribute to the problems we solve in class. You need to bring your laptop to class each day (and power adapter if needed). Our department has some laptops that you may borrow for the semester if you do not have a laptop. Most importantly, I expect you to ask questions if you do not understand something. Asking questions is the key to the intellectual life. Besides, there is no future in remaining confused. If you find yourself struggling, I expect you to attend office hours.

WORKING TOGETHER: I encourage you to form study groups. However, it is important that you submit your own work and that you do not allow another to do your thinking for you. You must acknowledge *and* understand the help you receive.

EXERCISES: Exercises are an ideal way to learn the skills and concepts that are the focus of this course; some will argue that it is the *only* way. You are expected to have the competence to solve every exercise at the end of each section of your text. I will identify a recommended subset of problems for you to start with each week. The point of doing exercises is to develop and strengthen skills. To reach a desired level of proficiency, some students will not need to do all the recommended exercises; others will have to do more. You must realize these are guidelines. You are required to submit homework exercises *and have them graded during office hours* if your cumulative average exam score is below 85%. If required to do so, plan to turn your solutions in well before the deadline as late work will not be accepted. Using the DrScheme environment, you will know for yourself if your solutions work or not. I will review a subset of the exercises in class.

PROJECT: During the latter portion of the semester, students will be given a more substantial programming assignment that will involve several phases of development. Each phase will have an associated deliverable that will be submitted for grading; together, the project deliverables will represent the primary component of the overall grade for the course. Failure to complete one or more installments of the project will significantly impact your grade in a negative manner.

EXAMS: There will be from six to twelve exams during the semester and a comprehensive final exam. Your lowest exam score will be ignored. Exams must be taken at the scheduled time unless arrangement is made with the instructor *in advance*.

ONLINE PORTFOLIO, READINGS AND DISCUSSION: You will create a portfolio webpage for yourself. The portfolio will contain your work from CS010 and future CS courses if you continue in with a major or minor. Also, a number of short readings will be assigned; you will be required to discuss these readings on Eureka forums.

ATTENDANCE: To learn a language, you must use it. In this respect, Scheme is no different from French. In the style of language immersion, we will go over numerous examples and you will write programs during class time. Therefore, attendance is required. If you have a reason for missing more classes than allowed by Westmont policies, discuss the matter with me *in advance*.

GRADING: Students will be evaluated on how well they master the skills of program design as demonstrated using the Scheme programming language. Both correctness and style are important and will be used to evaluate actual student work. A final weighted score will be based on the following: project (35%), collective exams (30%), exercises (20%), forum participation (10%), online portfolio (5%). The final weighted score, will yield a letter grade according to the standard scheme where 0.9 and above yields an A, 0.8 to 0.9 a B, and so on. I reserve the right to lower the thresholds for some or all of the letter grades; they will not be raised above the respective ten-percentile values.

ACADEMIC HONESTY: As in every area of life, I expect that you will conduct yourself honestly within the context of this class. You are expected to have read and agreed to the Academic Dishonesty policy as part of the general Westmont Academic Policies, as well as the specifics of the policy on Plagiarism. You should also find at the course webpage and carefully read the supplemental guidelines regarding working together and submitting work. Do not attempt to receive credit for work that is not your own without properly acknowledging sources via appropriate citations or references. You are encouraged to get help from your peers, but you must acknowledge such help (both received and given) and that you understand the issue on which help was received. Except where explicitly provided or allowed by the instructor, you must not use electronic copies of any portion of code for the exercises or project (unless distributed by the instructor). The consequences of violating the trust I implicitly extend to you will be according to the Westmont policy; but more serious will be the damage done to our academic relationship.