CS010 – Introduction to Computer Science I
Fall, 2011

Course Information

OVERVIEW: The course introduces students to the field of Computer Science via *programming*. We will use the Racket language, a derivative of *Scheme,* without assuming any prior programming experience. Using 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 Racket, 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. Reasoning abstractly)
- Implement solutions to problems using DrRacket to write programs.

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. As such, students will be able to :

> 1. Identify instances of abstract deductive reasoning about abstract objects or concepts (in the form of arguments, explanations, proofs, analyses, modeling, or processes of problem solving) and can distinguish premises from conclusions (or their analogues).
>
> 2. Construct an instance of valid deductive reasoning about abstract objects or concepts (in the form of arguments, explanations, proofs, analyses, modeling, or processes of problem solving).
>
> 3. Distinguish valid forms of deductive reasoning about abstract objects or concepts (in the form of arguments, explanations, proofs, analyses, modeling, or processes of problem solving) from invalid and/or fallacious forms of reasoning.

In the context of this class, abstraction takes several forms including functions, meta-functions, modeling, information hiding, hierarchical decomposition, and replication and repetition. Thus, we will place more emphasis on *inductive* than on *deductive* reasoning.

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: improve their ability to *think creatively and flexibly* (C3); improve their ability to *communicate effectively* (C2); start acquiring the *fundamental concepts and skills* of Computer Science (C1); and learn to identify initial *relationships between Christian faith*

and Computer Science (C4). 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 one of the Racket teaching languages. Such students will also have acquired the conceptual framework for continuing study in Computer Science.

LOCATION: Winter Hall, Room 216

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

TEXTS: [**required**, but freely available on-line] *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/F11

EUREKA WEBPAGE: http://eureka.westmont.edu/

OTHER MATERIALS: Laptop and USB flash drive

Instructor Information

INSTRUCTOR: Dr. Wayne Iba
OFFICE: Winter Hall 308
OFFICE HOURS: *posted on my web page, but available at other times by appointment*
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 should always bring your laptop to class as well as a USB flash drive. If you do not have a laptop computer, our department has a few loaners that you may borrow for the semester. Most importantly, I expect you to ask questions if you do not understand something. Asking questions is the key to the intellectual life. Use help sessions and office hours as needed.

WORKING TOGETHER: I encourage you to form study groups. However, it is important that you submit your own work and do not allow someone else to do your thinking for you. You must acknowledge *and* understand the help you receive. Acknowledgments *must* appear at the top of each submission using the provided form.

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***. Realize that these are guidelines and you should adjust for your own situation by practicing exercises until you master the relevant skills. I will be happy to provide additional exercises if you exhaust the problems found in the text. Start on assignments as soon as they are available and submit your solutions via Eureka well before the deadline as late work will not be accepted. Using the DrRacket environment and tests you create, you will know for yourself if your solutions work or not. Detailed feedback is available during office hours. I will review a subset of the exercises in class. Exercises will be evaluated according to their adherence to the *design recipe*. The lowest exercise score is dropped.

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 a significant 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 quizes during the semester and a comprehensive final exam. Your lowest quiz score will be ignored. Quizes must be taken at the time they are given 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 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, Racket 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 to miss class, discuss the matter with me *in advance*. More than two unexcused absences will negatively impact the grade. Missing more than four will result in an F for the course. Do not come to class if you are ill and potentially contagious, but do contact me prior to class.

GRADING: Students will be evaluated on how well they master the skills of program design as demonstrated using the Racket 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 carefully read the supplemental guidelines regarding working together and submitting work as well as my essay on cheating in computer science courses. 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. You should not share code in any form – printouts, screen shots, emails, or even looking at someone else's screen – except when *explicitly* instructed to do so by your Instructor. When asking for help from peers, you must formulate questions that can be answered without reference to specific code text. If you are asked to provide help, you must insist on such formulation of questions, and respond in kind. If in doubt, ask the Instructor. The consequences of violating the trust I implicitly extend to you will be according to my essay (subsuming Westmont's academic integrity policy); but more serious will be the damage done to our academic relationship.

STUDENTS WITH SPECIAL NEEDS: Students who have been diagnosed with a disability (learning, physical or psychological) are strongly encouraged to contact the Disability Services office as early as possible to discuss appropriate accommodations for this course. Formal accommodations will only be granted for students whose disabilities have been verified by the Disability Services office. These accommodations may be necessary to ensure your full participation and the successful completion of this course. Please contact Sheri Noble, Interim Coordinator of Disability Services (x6186, snoble@westmont.edu) as soon as possible.