Computer Science 062 – Spring 2013

Instructor & Texts

Instructors: **Kim Bruce**
222 Edmunds, x7-1866
Office Hours: MThF 2:00 - 3:00 p.m., W 11 to noon, & by appt.
kim@cs.pomona.edu
www.cs.pomona.edu/~kim/

Kevin Coogan
221 Edmunds, x7-8651
Office Hours: MW 3:00-4:00 p.m., Th 11:00-noon. & by appt.
kcoogan@cs.pomona.edu
www.cs.pomona.edu/~kcoogan/

Lectures: MWF 10:00 - 10:50 a.m., Edmunds 114

TAs: Sam Konowich, Kim Merrill, Sam Posner, Dominick Reinhold, and Andrew Russell
TA hours: Thursday, Sunday, and Monday evenings, 8:00-10:00 p.m.

Labs: T 1:15 p.m. - 2:30 p.m., T 2:45 p.m. - 4:00 p.m. in Edmunds 229

Texts: Duane Bailey, *Java Structures*, sqrt(7) edition, 2007. The book is available online.
Mark Allen Weiss, C++ for Java Programmers, Pearson Prentice Hall, 2004,
ISBN 9780139194245. Highly recommended.

Course web page: www.cs.pomona.edu/classes/cs062/

Overview

This course couples work on program design, analysis, and verification with an introduction to the study of data structures that are important in the construction of sophisticated computer programs. Because we will be interested in studying more modern techniques for designing and implementing efficient computer programs, we will primarily be using the object-oriented programming language, Java, though later in the course you will learn to develop programs in C++ as well. We will see that the object-oriented style of programming is extremely useful in designing large, complex programs and supporting reusable software.

Students will be expected to write several programs, ranging from very short programs to more elaborate systems. Since one of our goals in this course is to teach you how to write large, reliable programs composed from reusable pieces, we will be emphasizing the development of clear, modular programs that are easy to read, debug, verify, analyze, and modify.

Equally important is the ability to analyze programs for correctness using big-"O" notation. This will help us evaluate the trade-offs in different choices of algorithms and data structures.

We assume that all students enrolled are comfortable writing small to medium-sized programs (< 10 pages of code with several interacting classes) in Java. The knowledge assumed is generally equivalent to that of CSCI 051 as offered at either Pomona or CMC or the Computer Science advanced placement exam. If you have any doubts as to whether your programming experience is sufficient for this course or if you have never programmed in Java, please see us as soon as possible.

By the end of this course you should have a good understanding of the object-oriented design, coding, and debugging of programs in Java and C++, and have a good understanding of how one might analyze programs for correctness and efficiency. In particular, you will understand the trade-offs involved in selections of different data structures and algorithms to solve computational problems.

This course is a prerequisite for most upper level Computer Science courses. It may be taken either before or after CSCI 052.

Lectures and Readings

Students should consult the on-line version of the course syllabus (see URL above) regularly to see the most current version of the schedule of topics and readings.

All reading assignments are from the text. Students should come to class having completed the indicated readings for the day. You should attempt to work all the problems at the end of each section as you are reading. Chapter review problems will be assigned during each lecture. While chapter review problems will not be turned in, you should complete them before the next lecture after they are assigned. Many of these will show up in the regular quizes during the first five minutes on Friday mornings, and may also appear on the midterm or final. Problems assigned in association with lecture n should be completed before lecture n+1. Answers are in the back of the book for most problems assigned.

Programming Assignments and Laboratories

Labs for this course will be held on Wednesday afternoons from 1:15 p.m to 2:30 p.m. and from 2:45 p.m. to 4 p.m. in 229 Edmunds. The room is equipped with iMac computers.

Attendance at these lab sessions is **mandatory**. Please arrive well prepared for the lab or you will waste your time and ours.

Unlike in CS 51 at Pomona, lab work will generally be distinct from the weekly programming assignments, though the lab work will generally be relevant to those longer assignments. Instead we will often use lab time to introduce you to new software tools and techniques that require more hands-on experience to understand. You will usually need to submit your results from the lab by end of the lab period.

There will be two types of weekly programming assignments: *individual programs* and *team programs*. All programs assigned during the semester should be completed following the guidelines in the *Academic Honesty Policy* described below.

There will be 10 to 12 weekly programs due. All programs will be graded on design, documentation and style, correctness, and efficiency. The elements of a good program are very much like the elements of a good paper. It must be correct, but it should also be written in a style that is clear and elegant. You will receive written comments on all of your programs.

Weekly assignments will generally be due on Monday evenings at 11:59 p.m. There will be a penalty assessed of 3^n % for a program that is n days late. Programs will not be accepted more than four days late. It is usually better to turn in a correct and well-documented program one or two days late than a non-functioning or non-documented program on time.

All assignments should be submitted electronically. The procedure will be explained in laboratory.

Wednesday	**Laboratory Title**	**Weekly Assignment**
Jan. 30	Eclipse & Silver Dollar	Graphic Silver Dollar Game
Feb 6	Timing Vector Additions	Word Frequency
Feb. 13	Analysis of Algorithms	Disk sort
Feb. 20	Two Towers	Compression
Feb. 27	Eclipse debugging	Calculator
March 6	JUnit	Darwin
March 13	Binary Trees	Darwin, cont.
March 27	Binary Search Trees	Hex-A-Pawn
April 3	Parallelism	Census Data
April 10	C++	Priority Queue in C++
April 17	Linked Lists in C++	Animals in C++: Part 1
April 24	C++ Memory Management	Animals in C++: Part 2
May 1	Dijkstra's algorithm	Graph Algorithms in C++: Part 2
May 8	No lab!	

Exams

There will be one in-class midterm exam plus a scheduled final exam. There will also be quizes every week except the first week of classes. The quizes will be during the first five minutes of class. No make-ups will be allowed, but the two lowest scores will be dropped.

- *Midterm examination*: Monday, March 11, in class.

  Two sample exams: midterm 1 and midterm 2 from a similar course I taught several years ago are available. The exams were for a period longer than 50 minutes and the second exam covers material beyond what will be covered on our exam. Nevertheless, you may find them useful in determining the kinds of questions that I like to ask.

- *Final examination*: Monday, May 13, at 9 a.m.

  A sample final exam is available. That course covered no C++, as well as having some other minor variations in coverage, but it should give you a sense as to the kind of questions that might be asked. While the exam coverage is comprehensive, most of the focus will be on material since the midterm.

Grading Summary

Lab & Weekly Programming Assignments:			35%
Exams:	Total:		55%
	Midterm Exam:	25%	
	Final Exam:	30%	
In-lab exercises & quizzes:			10%
Total:			100%

Collaboration & Academic Honesty Policy

We highly encourage students to get together in small groups to go over material from the lectures and text, work practice problems from the text, study for exams, and to discuss the general ideas and approaches to laboratory assignments. However, work to be turned in, including programming assignments, must be done according to the rules in effect for that assignment.

When a program is assigned, we will identify it as either an "individual" or a "team" program. The academic honesty policy applies differently to each with respect to collaboration or assistance from anyone other than the mentors or instructors:

This course has two kinds of programming assignments, individual programs and team programs.

Individual Programs Individual programs are expected to be the work of the individual student, designed and coded by him or her alone. Help locating errors is allowed, but a student may only receive help in correcting errors of syntax; help in correcting errors of logic is strictly forbidden. Guideline: Assistance from anyone other than the TAs or instructors in the design or coding of program logic will be considered a violation of the academic honesty policy.

Team Programs Team programs are programs to be worked on in teams of two or more students. You are allowed to discuss team programs with your partners, but work with others is otherwise restricted by the appropriate rules above. Guideline: Any work that is not from your team is considered a violation of the academic honesty policy.

If you do not understand how the academic honesty policy applies to a particular assignment, consult with us. When in doubt, credit the people or sources from whom you got help. This also goes for any help obtained via the Internet. If you are ever unsure about what constitutes acceptable collaboration, just ask.

Failure to abide by these rules is considered plagiarism, and will result in severe penalties. Violations are easy to identify and will be dealt with promptly. The first offense typically results in failure in the course. A second offense is automatically referred to the College's Board of Academic Discipline. See the Academic Honesty Policy in the Student Handbook for further information. Please do not put us, yourself, or anyone else in this unpleasant situation.