# Computer Science 051 – Spring 2006

## Instructors & Text


Instructors: **Kim Bruce** (lectures)
212B Millikan, x7-1866
kim@cs.pomona.edu
Office Hours: MThF 1:15-2:15 p.m., W 9 a.m. - 9:50, & by appt.
**Joe Vanderwaart** (labs)
263 Andrews, x7-0486
joev@cs.pomona.edu
Office Hours: M 1 - 3 p.m., T 4 - 5 p.m., & by appt.


Lectures: MWF 10:00 - 10:50 a.m., Millikan 213
Lab: W 1:15 p.m. - 4:00 p.m., Andrews 257
Text: *Java: An Eventful Approach* by Kim B. Bruce, Andrea Pohoreckyj Danyluk, and Thomas P. Murtagh, Prentice Hall, 2006.
Course web page: www.cs.pomona.edu/classes/cs051/
Instructors' web pages: Bruce: www.cs.pomona.edu/~kim/
Vanderwaart: www.cs.pomona.edu/~joev/

## Overview

Computer Science 051 is an introduction to algorithm development emphasizing object-oriented design. Algorithms will be implemented as programs in the Java programming language. We will introduce data structures and recursion as tools to construct correct, understandable, and efficient algorithms. These topics will be developed further in CSC 052 and more advanced computer science courses. For those who wish a good introduction to the science of computing. we highly recommend the combination of CSC 051 and CSC 052, which includes an introduction to a broad array of topics in computer science.

By the end of this course you should have a good basic understanding of how to develop (design, code, and debug) medium-sized programs in Java, and have a basic understanding of how one might analyze programs for correctness and efficiency.

This course is a prerequisite for all upper level Computer Science courses. In Computer Science 051 *we do not assume that you have had any previous computer programming experience.* If you have had extensive previous experience you might consider CSC052. Please discuss this with your instructor *immediately* if you feel you fall into this category.

## Lectures and Readings

The schedule on the following two pages shows the topics to be covered at each class meeting during the semester. 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.

| Lecture | Date | Topic | Reading | Hmwk assigned |
|---|---|---|---|---|
| 1. | Jan. 18 | Fundamentals & Graphics | Ch 1 | 1.9.3 |
| 2. | Jan. 20 | Event Handling & Numbers | Ch 2, 3 | |
| 3. | Jan. 23 | Conditionals | Ch 4 | |
| 4. | Jan. 25 | More conditionals | Ch 4 | |
| 5. | Jan. 27 | Primitive types & classes | Ch 5 , 6.1-6.3 | |
| 6. | Jan. 30 | Class declarations | Ch 6.4-6.7 | |
| 7. | Feb. 1 | Control Structures | Ch 7 | |
| 8. | Feb. 3 | Scope & Active Objects | Ch 8, 9.1-9.2 | |
| 9. | Feb. 6 | Active Objects | Ch 9.3-9.5 | |
| 10. | Feb. 8 | More active objects | Ch 9.6-9.9 | |
| 11. | Feb. 10 | Interfaces | Ch 10 | |
| 12. | Feb. 13 | GUI components | Ch 11.1-11.3 | |
| 13. | Feb. 15 | More GUI components | Ch 11.4-11.6 | |
| 14. | Feb. 17 | More GUI components | Ch 11.7-11.8 | |
| 15. | Feb. 20 | Recursion | Ch 12.1 | |
| 16. | Feb. 22 | More recursion | Ch 12.1 | |
| 17. | Feb. 24 | Even more recursion | Ch 12.2 | |
| 18. | Feb. 27 | For loops | Ch 13 | |
| 19. | Mar. 1 | Arrays | Ch 14.1-14.5 | |
| 20. | Mar. 3 | More Arrays | Ch 14.6-14.8 | |
| 21. | Mar. 6 | Multi-dimensional Arrays | Ch 15.1-15.2.2 | |

| Lecture | Date | Topic | Reading |
|---|---|---|---|
| 22. | Mar. 8 | *Midterm* | |
| 23. | Mar. 10 | More multi-dimensional Arrays | Ch 15.2.3-15.3 |
| | Mar. 13-17 | **Spring Break** | |
| 24. | Mar. 20 | Strings | Ch 16.1-16.2 |
| 25 | Mar. 22 | More Strings | Ch 16.3 |
| 26. | Mar. 24 | More Strings | Ch 16.4 |
| 27. | Mar. 27 | Inheritance | Ch 17.1-17.5 |
| 28. | Mar. 29 | More Inheritance | Ch 17.6-17.7 |
| | Mar. 31 | **College Holiday - Chavez Day** | |
| 29. | April 3 | Exceptions | Ch 18 |
| 30. | April 5 | Streams | Ch 19.1-19.3 |
| 31. | April 7 | More Streams | Ch 19.4 |
| 32. | April 10 | More Streams | Ch 19.5 |
| 33. | April 12 | Networking | Ch 19.6 |
| 34. | April 14 | Lists | |
| 35. | April 17 | Linked Lists (cont) | |
| 36. | April 19 | Stacks & Queues | |
| 37. | April 21 | Design | Ch 21 |
| 38. | April 24 | Searching | Ch 20.1 |
| 39. | April 26 | Sorting | Ch 20.2-20.3.4 |
| 40. | April 28 | Sorting | Ch 20.3.5-20.5 |
| 41. | May 1 | Concurrency | |
| 42. | May 3 | Summary | |

## Programming Assignments and Laboratories

There will be two types of programming assignments: *laboratory programs* that, as a unit, count for 15% of your final grade, and two *test programs* that make up 40% of the final grade. Additional information about these programming assignments can be found on page 5. Test programs are treated as take-home exams, with no assistance allowed. All programs assigned during the semester should be completed following the guidelines in the *Academic Honesty Policy* described below.

Laboratory programs will not be accepted after the due date, but in computing your grade the lowest score on submitted assignments will be dropped. Late test programs will be accepted, but with a penalty of 10% per day. Labs for this course will be held on Wednesday afternoon from 1:15 p.m to 4:00 p.m. in 257 Andrew. The room is equipped with "dual-boot" Dell computers that run both Windows and LINUX. We will be using the free Eclipse IDE (Interactive Development Environment) to create and run all of the programs in the course (sample programs, lab exercises, and homework assignments). Because the computers run both Windows and LINUX you may have to reboot your computer at the beginning of lab so that it is running LINUX.

The purpose of the lab sessions is to provide a time during which your instructor can actively assist you in the development of *laboratory programs*. Attendance at these lab sessions is **mandatory**. The initial lab period will familiarize you with the use of LINUX and course software. Remaining labs will be handed out in advance, and you will be expected to have planned your approach to the assignment before the beginning of lab. The lab session will then be used to develop the program and perform related exercises. You will submit your laboratory programs electronically. The procedure will be explained in laboratory. *Laboratory programs will generally be due by 11:00 P.M. on the Thursday night after your lab session.*

| Wednesday | Laboratory Title |
|---|---|
| Jan. 18 | Intro. to Java |
| Jan. 25 | Event Handling with Conditionals |
| Feb. 1 | Defining Classes |
| Feb. 8 | Parameters |
| Feb. 15 | Loops and Simple Threads |
| Feb. 22 | GUI Practice |
| **Thursday, Feb. 23** | **Test Programs 1 distributed** |
| **Friday, March 3** | **Test Programs 1 Due at 4:00 P.M.** |
| **Wednesday, March 8** | **Midterm Exam in class** |
| March 8 | Recursive Definitions |
| March 22 | Arrays |
| March 29 | Two-D Arrays |
| April 5 | Strings |
| April 12 | Streams |
| **Thursday, April 13** | **Test Program 2 distributed** |
| April 19 | Linked Lists |
| April 26 | Work on test program |
| **May 3** | **Test Program 2 Due, 4:00 P.M.** |

## Exams

There will be an in-class midterm (worth 20% of your grade). There will also be a scheduled final exam (counting 25%).

*Midterm examination*: Wednesday, March 8, in class.

*Final examination*: Wednesday, May 10, at 9 a.m.

## Grading Summary

| | | | |
|---|---|---|---|
| **Programs:** | | | 55% |
| | Laboratory Programs | 15% | |
| | Test Programs | 40% | |
| **Exams:** | | | 45% |
| | Midterm Exam: | 20% | |
| | Final Exam: | 25% | |
| **Total:** | | | 100% |

## Collaboration & Academic Honesty Policy

I highly encourage students to get together in small groups to go over material from the lectures and text, work 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 independently, unless I explicitly designate an assignment as one in which collaboration is allowed. As expained in the student handbook, this means that the work you turn in must represent only your own work. It must not be based on help from others or information obtained from sources other than those approved by the instructor (e.g., the text, web pages linked from the course web page, and materials provided in lecture). Effective learning is compromised when this is not the case.

Accordingly, you should never read or copy another student's code or solutions, exchange computer files, or share your code or solutions with anyone else in the class until after the assignment is due. Under no circumstances may you hand in work done by, or in collaboration with, someone else under your own name, with the exception that you may freely use any code that I provide to you or code from the textbook, as long as you cite this code as coming from the instructor or the book. Additionally, the student mentors are allowed to help you with your code.

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

**Test Programs.** Any assignment designated as a major program is to be treated exactly as a take-home, open-book test with respect to rules for obtaining assistance. You are allowed to read your textbook, class notes, and any other source approved by your instructor. You may not consult anyone other than your instructor. The instructor encourages the asking of questions, but reserves the right not to answer, just as you would expect during an exam. Guideline: Any work that is not your own is considered a violation of the academic honesty policy.

**Laboratory Programs.** Laboratory 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 mentors or instructors in the design or coding of program logic will be considered a violation of the academic honesty policy. The only time it is acceptable to look at another student's code is to help them identify an error in their program. You may not provide them with suggested code to correct it nor may you copy their code in your program. You may point out similar examples from the text or lecture notes that help them correct the error.

If you do not understand how the academic honesty policy applies to a particular assignment, consult with me. When in doubt, credit the people or sources from whom you got help. This also goes for any help obtained via the Internet. You will not lose any points for acknowledging help obtained where the rules for assistance are unclear. 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 me, yourself, or anyone else in this unpleasant situation.