

CSCI-2300: Data Structures and Algorithms

Sections 1, 2, 3, 7

Spring 2002

1 Course Information

Instructor: Prof. Srinivas Akella
Email: sakella@cs.rpi.edu
Office: Amos Eaton 112, x8770
Office hours: Tuesday 2:00-3:30pm, Thursday 2:00-3:30pm

Lectures: Tuesday, Friday 10:00am–11:20am
Classroom: Sage 3303
Prerequisites: Computer Science II (CSCI-1200) and Calculus I (MATH-1010)

Course web page: Course announcements and information will be available at <http://www.cs.rpi.edu/~sakella/dsa>

Textbooks:

Required textbook: *Data Structures and Algorithm Analysis in C++, second edition*. Mark A. Weiss, Addison Wesley, 1999.

Recommended textbook: *The C++ Programming Language, third edition*. Bjarne Stroustrup. Addison Wesley, 1997. You may choose a different recent C++ reference book if you wish.

2 Course Overview

We will study intermediate and advanced data structures, computer algorithms, and the analysis of both using techniques from discrete mathematics. Data structures and algorithms form a major component of any software system. When building such a system, a skilled computer scientist must make intelligent decisions about alternative techniques, choosing from existing data structures and algorithms or designing his/her own when necessary. While we will concentrate on the theoretical design and analysis of data structures and algorithms, we will reinforce the theory with working examples, laboratories, programming projects, and use of the C++ standard library.

Classes and Class Preparation

This is a 4 credit course that meets in two 80-minute class sessions and one 2-hour lab session each week. Class will consist of lectures that will expand on, explain, and reinforce the reading

material. The reading material will be selected sections from the Weiss text. By reading in advance, students will be prepared for class and will be better able to understand the material presented.

Labs

There will be 12 lab sessions during the semester. These will emphasize programming aspects of data structures and algorithms. Labs will be graded, but this grading will be lenient. You **MUST** go to your assigned lab section in order to receive credit for that day's lab. Labs will count 10% of your semester grade.

Section 1	Wednesday 8:00-9:50am	Sage 3101
Section 2	Wednesday 10:00-11:50am	Sage 3101
Section 3	Tuesday 6:00-7:50pm	Sage 3101
Section 7	Wednesday 4:00-5:50pm	Sage 3101

Programming Projects

There will be four programming projects given throughout the semester. Programs must be written in C++ and should use the standard library as needed. Students will have at least two weeks to do each project. Projects must be submitted by 11:59:59 pm on the scheduled due date. Submission will be by file transfer to a specified RCS directory. Programming projects will count 30% of the semester grade.

Lateness Policy: Each student will be given three days (whole or partial) of grace for late programs. Use these late days carefully. Once you have exhausted these days, **late programs will not be accepted** without a written excuse from the Dean of Students' office. This includes late days for illnesses, plant trips, etc.

As an example, if you submit your 1st program 26 hours late, you will have used two late days and have only one day left. If you submit your 2nd program 5 hours late, you will have used your last late day. If you then submit your 3rd program 1 minute late, it will not be accepted. **USE YOUR LATE DAYS WISELY, IF AT ALL.**

Written Homeworks

There will be five written homework assignments spaced throughout the semester, supplemented with occasional exercises. These will count 15% of the semester grade. You will typically have at least a week to complete each homework. **No late homeworks will be accepted.**

Exams

There will be two in-class exams during the semester and a final exam. Exams will be closed-book and will be based on material covered in class, assigned readings, labs, projects, and homeworks. The weight of the final will vary for each individual student, depending on how well he or she does on the final. If a student's grade on the final is worse than each of the two in-class exams, then all three exams are worth 15%. If a student's grade on the final is better than at least one of the in-class exams, then the final is worth 25%, the worst in-class exam is worth 5%, and the other in-class exam is worth 15%. No makeup exams will be scheduled; in exceptional circumstances, prior arrangements must be made with the instructor.

Summary of Requirements and Grading:

Laboratories	10%
Programming Projects	30%
Homeworks and exercises	15%
Exams	45%

The maximum lower bound cutoffs for A, B, C, and D grades will be 90%, 80%, 70% and 60%, respectively. Lower cutoffs may be established at the end of the semester when assigning grades. Exam or project grades that result in a low class average will be curved upwards.

Class Schedule

Here is a schedule of the coverage of course material. We will stick to the schedule closely. Detailed reading assignments will be given in class.

Class(es)	Date(s)	Topic
1-3	1/15-1/22	C++ Review (Ch 1.4-1.6, App A)
4,5	1/25-1/29	Linear structures (Ch 3)
6-9	2/1-2/12	Algorithm analysis, induction, recursion (Ch 1.1-1.3, 2)
11-14	2/22-3/5	Trees (Ch 4)
15	3/8	Hashing (Ch 5)
16-17	3/19-3/22	Priority queues (Ch 6)
18-19	3/26-3/29	Sorting (Ch 7)
20,22-25	4/5-4/19	Graphs and graph algorithms (Ch 9)
26-28	4/23-4/30	Advanced material (TBA)

Labs will be held every week of the semester except the weeks of February 18-22, March 11-15 (spring break), April 8-12, and April 29-May 3.

Important Dates

Here is a tentative schedule of homework and project due dates and exam dates.

Class(es)	Date(s)	Event
5	1/29	Homework 1 due
	2/6	Project 1 due
9	2/12	Homework 2 due
10	2/15	In-class exam 1
	3/4	Project 2 due
15	3/8	Homework 3 due
18	3/26	Homework 4 due
	3/28	Project 3 due
21	4/5	In-class exam 2
24	4/16	Homework 5 due
	4/26	Project 4 due

3 Academic Honesty

Academic integrity is a problem on programming assignments. Students naturally want to work together, and it is clear they learn a great deal by doing so. Getting help is often the best way to interpret error messages and find bugs, even for experienced programmers. In response to this, the following rules will be in force. Students are allowed to work together in high-level design of algorithms (i.e., pseudocode), in interpreting error messages, in finding bugs, but NOT in writing code. Students may not share code, copy code, or discuss code in detail (line-by-line or loop-by-loop) while it is being written or afterwards. Students may not “show” their code to other students as a means of “helping them”. Students may not leave their code (either the electronic versions or the printed copies) in publically accessible areas. Shared or copied code is easy to spot manually and is easily detected using a variety of software tools. Students caught illegally collaborating in writing code or violating the above rules will receive a 0 on the assignment plus a 5 percentage point penalty on their semester grade and a report to the Dean of Students office. Students caught a second time will receive an F in the course and will be reported to the Dean of Students office.

Copying or using disallowed materials during an exam is cheating, of course, and will result in a 0 on the exam, plus a 5 percentage point penalty on the semester grade and a report to the Dean of Students office.

Students are allowed to work together on homeworks and labs. Students must however write their homeworks and labs individually. Again, copying on this course work is not allowed and will result in a 0 for the submission plus a 5 percentage point penalty on the semester grade and a report to the Dean of Students office.

Any student violating these academic honesty rules for a second time will receive an F in the course and will be reported to the Dean of Students office.

Refer to the Rensselaer Handbook, which defines various forms of academic dishonesty and procedures for responding to them. Students found in violation of academic honesty policies may receive a failing grade for the course.