CS 361  Data Structures and Algorithms I  Fall 2001

Syllabus

Instructors:
Henry Shapiro, Professor of Computer Science
Class Meeting Time: MW 9:00–10:15 (Woodward Hall 149)
Office: FEC 319, 277–3052
Office Hours: MW 12:30–4:00
e-mail: shapiro@cs.unm.edu
web address: www.cs.unm.edu/~shapiro

Teaching Assistant: Bryan Cheng
Recitation Section 1: Th 9:30–10:20 (Dane Smith Hall 120)
Recitation Section 2: F 9:00–9:50 (Woodward Hall 149)
Office: 301A, 277–3394
Office Hours: TTh 2:00–5:00, F 10:00-12:00 and 1:00-3:00
e-mail: bryan@cs.unm.edu or bcheng@unm.edu

2001–2003 Catalog Data:
CS 361L. Data Structures and Algorithms I. (3)
An introduction to data structures and algorithms and the mathematics needed to analyze their time and space complexity. Topics include $O(\cdot)$ notation, recurrence relations and their solution, sorting, hash tables, priority queues, search trees (including at least one balanced tree structure), and basic graph representation and search. Course includes programming projects.
Prerequisites: 201, 251L, and Math 163L.


Topics: The order in which the topics will be presented is not the order in which they are listed below and the mathematics component (items 1–3) will be integrated into the data structures/algorithm component (items 4–8).

1. Fundamental mathematics: Logarithms and exponentials, series and limits, binomial coefficients. [This is review material.]
2. $O(\cdot)$, $\Theta(\cdot)$, and $\Omega(\cdot)$ notation, including their formal definition and how to analyze the complexity of iterative programs. Worst- and average-case analysis.
3. Solution of inhomogeneous linear recurrences with constant coefficients and how they can be used to analyze the time complexity of recursive programs.

4. Hash tables, including the analysis of the average-case running time for insertion and find for uniform hashing.

5. Review of binary trees and binary search trees. Analysis of the average-case behavior of binary search trees. Balanced binary search trees (AVL trees will be discussed in detail; red-black trees will be presented if time permits).

6. Priority queues implemented as classical binary heaps, including DECREASEKey, and HEAPIFY and the analysis of the various operations.

7. Sorting, including mergesort, quicksort and heapsort and their analysis. Lower bound of \(O(n \log n)\) for sorting.

8. Basic graph representation (adjacency matrix and adjacency lists) and basic search algorithms including depth-first and breadth-first search. Efficiency of algorithms when different representations are used. (This topic covered if time permits.)

Assignments, Examinations, and Grading: The programming in this course will be part of a larger term project. The term project will be a comparison of the performance of two data structures implementing the same ADT (abstract data type). This will involve three phases: (1) coding up the actual data structures, (2) designing an experiment that measures the performance of the two data structures—this will involve additional coding, as you will have to write various drivers and/or instrument your program to measure its performance, (3) writing a report detailing and explaining your findings. The majority of the grade will be allocated to phases two and three. The programming of the data structures themselves, while that may take a lot of time, is not the main thrust of the project: the thoughtfulness of your experiment and the way you write it up is what the project is really about.

There will be one midterm and a final. There will also be some homework assignments that stress the mathematical aspects of the course. The exams will likely have both an in-class and take-home portion. The in-class portion of the final examination will be held on Wednesday, December 12 from 3:00–5:00. If there is a take-home portion to the final it will be due on Friday, December 14 at 12:00 noon. Don’t make Christmas travel plans that interfere with these dates.

The following table shows the percentage of your grade that will be...
assigned to each aspect of this course—these percentages are subject to change and should be used only as a guideline.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Percentage Of Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homework Assignments</td>
<td>10%</td>
</tr>
<tr>
<td>Term Project</td>
<td>40%</td>
</tr>
<tr>
<td>DELETE in a binary search tree</td>
<td>5%</td>
</tr>
<tr>
<td>[This is a warmup exercise.]</td>
<td></td>
</tr>
<tr>
<td>AVL trees</td>
<td>15%</td>
</tr>
<tr>
<td>Additional data structures, experiment and report</td>
<td>20%</td>
</tr>
<tr>
<td>Midterm Examination</td>
<td>20%</td>
</tr>
<tr>
<td>Final Examination</td>
<td>30%</td>
</tr>
</tbody>
</table>

Note: *This is not a programming class*, though there is a considerable programming component. More than 70% of your grade is based on your mastery of the theoretical and mathematical aspects of the course. You should make sure that you allocate sufficient time to study the material of the course and not think that all your effort should be put into the programming component.