Fall 9-1-2000

CS 531.01: The Design and Analysis of Algorithms

Alden H. Wright
University of Montana - Missoula, alden.wright@umontana.edu

Computer Science 531, Design and Analysis of Algorithms
Fall semester, 2000

Course description and Syllabus

Instructor: Alden Wright.

Office: 407 Social Science
Hours: 1:10 a.m. – 2:30 p.m. on Mondays, Wednesdays, Fridays
Feel free also to drop in to see if I am in my office
(and not occupied talking to someone else or getting ready for class).
Phone: 243–4790
E-mail: *wright@cs.umt.edu*
During weekdays, I can usually respond quickly to e-mail messages.

Prerequisites: Math 225 or equivalent background in Math. Programming background similar to that obtained by taking CS 132. If you lack these prerequisites, you must see me to discuss whether or not you should take the course.

Required texts:

- **Introduction to Algorithms** by Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. (McGraw Hill, 1990).

Evaluation:

- Readings as assigned with the homework.
- Homework exercises, handed out every other class period, due two class periods later. All or most of the homework will be paper-and-pencil work, perhaps assisted by Maple.

  I will try to hand out answers to each homework soon after it is due, so late homework will not normally be accepted.
- Two hour exams.
- A final examination.

Objectives:

At the completion of this course, the student should be able to:

- Read and write algorithms in pseudocode.
- Understand and do exercises (including proofs) on the asymptotic growth of functions.
- Use the substitution, iteration, and master methods to solve recurrences.
- Use summations to analyze algorithms.
- Use recurrences to analyze algorithms.
- Prove that the lower bound for sorting based on comparisons is $\Omega(n \log n)$.
- Understand and analyze randomized quicksort, counting sort, and radix sort.
- Use the recursion/induction paradigm to design algorithms.

- Use the divide and conquer paradigm to design algorithms.
- Use the dynamic programming paradigm to design algorithms.
- Use the greedy paradigm to design algorithms.
- Apply amortized analysis to analyze algorithms.
- Understand and do exercises (including proofs) on the theory of NP completeness.
- Understand how to prove the correctness of an algorithm.
- Understand a number of important algorithms.

Grading plan:

Weekly homework:	35%
Exam:	65%

Doing the weekly homework is necessary to succeed in this course. In a sense, it counts far more than merely 35%.

Students will be required to sign the statement on collaboration and cheating.

Incompletes will be given only for doctor-verified illness, death in the imediate family, and other reasons of similar consequence. The last day for partial refunds is Feb. 12, and the drop deadline is March 8.

Syllabus (for the course Fall 2000)

We will cover "mathematical foundations" since students find this difficult, and we will probably cover it in more depth than you have seen in previous courses.

Syllabus for Fall 2000:

- Introduction
- Growth of Functions
- Summations
- Recurrences
- Quicksort
- Sorting in Linear Time
- Dynamic Programming
- Greedy Algorithms
- Amortized Analysis
- NP-Completeness

Some optional topics if there is time:

- Randomized algorithms.
- Distributed algorithms.