Computer Science / Mathematics 340
General Information

Instructor: *Steven Lindell* Office: Link 308
(610) 896-1203 slindell@haverford.edu

Title & Description: *Analysis of Algorithms* (cross-listed in Mathematics and Computer Science) Algorithmic design and analysis, sorting, searching, data structures, graph algorithms – including a partial introduction to parallel algorithms. The emphasis will be on correctness and complexity.

Class schedule & room: Lecture MW 12:30-2:00 Discussion F 12:30-2:00 both in Link 310

Consultation hours: MW 2:00-3:00 and also by appointment. You can even call me at home using NetMeeting which allows sharing written information.

Text: *Introduction to the Design and Analysis of Algorithms*, Anany V. Levitin, Villanova University, 2nd edition ©2007.

Supplementary reading: *Introduction to Algorithms* Corman, Leiserson, Rivest
Computer Algorithms Sara Baase © 1988.

Prerequisites: Computer Science 206 (or comparable experience programming).

Grading: (discuss 50% rule for points)

Homework	30%	(includes class participation)
Midterm	30%	(one take-home examination)
Final	40%	(cumulative)

Homework: Weekly exercises -- late homework problems will be downgraded approximately 1/3 per day. Your work should be neatly and clearly presented, as these are *very* important in the grading (and learning) process. The solutions you hand in should be understandable to someone who knows the material, but not necessarily your approach. Not doing (or at least trying) all the homework is a good recipe for a poor (or failing) grade. All late homework must be complete before any exam.

Rules and regulations: Everything turned in for a grade must be your own work, although collaboration on problems is strongly encouraged (especially working in groups). Spoken and scribbled *ideas* on how to solve homework problems may be exchanged, but *not* detailed written solutions. You must write up your own solutions and should acknowledge your collaborators. You may use the Internet *only if the problem's hint suggests that you do so*, but you still must cite the source and put it into your own words. There is no assistance or help allowed on the examinations.

Special accommodations: Students who think they may need accommodations in this course because of the impact of a disability are encouraged to meet with me privately early in the semester. Students should also contact Rick Webb, Coordinator, Office of Disabilities Services (rwebb@haverford.edu, 610-896-1290) to verify their eligibility for reasonable accommodations as soon as possible. Early contact will help to avoid unnecessary inconvenience and delays.

Syllabus

(from book preface)

Lecture	Topic	Sections
1	Introduction	1.1-1.3
2, 3	Analysis framework; O, Θ, Ω notations	2.1-2.2
4	Mathematical analysis of nonrecursive algorithms	2.3
5, 6	Mathematical analysis of recursive algorithms	2.4-2.5+App. B
7	Brute-force algorithms	3.1-3.2(+3.3)
8	Exhaustive search	3.4
9-10	Divide-and-conquer: mergesort, quicksort, binary search	4.1-4.3
11	Other divide-and-conquer examples	4.4 , 4.5, or 4.6
12-14	Decrease-by-one: insertion sort, DFS & BFS, topological sorting	5.1-5.3
15	Decrease-by-a-constant-factor algorithms	5.5
16	Variable-size-decrease algorithms	5.6
17-19	Instance simplification: Presorting, Gaussian elimination, balanced search trees	6.1-6.3
20	Representation change: heaps and heapsort or Horner's rule and binary exponentiation	6.4 or 6.5
21	Problem reduction	6.6
22-24	Space-time tradeoffs: string matching, hashing, B-trees	7.2-7.4
25-27	Dynamic programming algorithms	8.1-8.4
28-30	Greedy algorithms: Prim's, Kruskal's, Dijkstra's, Huffman's	9.1-9.4
31-33	Iterative improvement algorithms	10.1–10.4
34	Lower-bound arguments	11.1
35	Decision trees	11.2
36	P, NP, and NP-complete problems	11.3
37	Numerical algorithms	11.4 (+12.4)
38	Backtracking	12.1
39	Branch-and-bound	12.2
40	Approximation algorithms for NP-hard problems	12.3