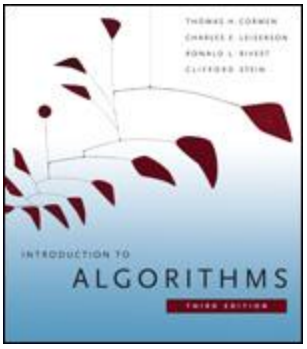
COMP 550–001: Introduction to Scientific Programming

Instructor: Jack Snoeyink, snoeyink@cs.unc.edu,
Lectures: TTh 11:00 – 12:15, FP Brooks Bldg 007
Office hours: Sitterson 329, 962-1969, M 12:00-1, Tu 3:30-4:30pm, F 11-12,
or by appointment

Welcome: This is the class in the undergraduate Computer Science curriculum that most celebrates clever ideas, as we need them to solve problems under resource constraints. It will, I hope, help you to hone your problem solving skills, and your ability to express solutions precisely. The goal is not to memorize but to understand, building skills that will transfer to many problem-solving situations. And it is fun; a computer scientists' "Glass Bead Game" (see the book by Herman Hesse; also Neil Stephenson's Anathem).



Textbook: "Introduction to Algorithms" by CLRS (Cormen, Rivest, Leiserson, and Stein) is widely used for both undergraduate and graduate algorithms classes. Either the 2nd or 3rd edition will work; the 3rd has improved several chapters, so be sure you get it if you are buying new. Solutions of many exercises can be found at http://mitpress.mit.edu/algorithms/.

Web site: We use the Blackboard web site, http://blackboard.unc.edu, for access to handouts, email, discussion boards, and an on-line gradebook. You need your ONYEN to log in to Blackboard. (See http://onyen.unc.edu if you need an onyen. Forgot your password? Bring your ID to the basement of Wilson.)

Evaluation Grades will be based on participation and keeping up with regular exercises (20%), three monthly "midterms" (14%,18%,18%) and a final exam (30%). Work may be turned in in class or under Assignments on blackboard. My questions are usually hard, so I curve everything.

I will often ask you to write out something or do a small quiz in class; most of these are for feedback to me, so I can gauge understanding of a topic. If I ask you to put your name on it, then I will use it to keep track of participation, but otherwise those will not be graded.

Exercises will come from the book, and may or may not have solutions online. I plan to just check them for understanding, but if I have to grade them to ensure that they serve their purpose of encouraging you to engage with the material from lectures and from the text, I will.

Midterms on Feb 4, Mar 4, and Apr 8 will be a combination of take-home and in-class problems.

Final exam is cumulative, and will also be a combination of take-home and in-class problems.

Cheat sheets: For any in-class portion of a test or exam, you may take a one-page, 8.5x11 in "cheat sheet" with whatever you want to write on it. (Algorithms, formulae, theorems, definitions, words of encouragement…) You must, however, write it yourself; photocopies or scans are only permitted if that is how you reduce larger writing to smaller size. (The process of preparing a sheet is usually more valuable than the sheet itself.)

Collaboration: Collaboration is encouraged on exercises and on the take-home parts of the midterms and final. I insist that whatever you hand in must be your own writing/typing. (In class portions will test whether you understand what you turn in for your take-home portions).

Good scholarship requires that resources and collaboration be acknowledged. Thus, if you collaborate on the solution of a problem set, I expect that you list your collaborators at the top of the page. Collaboration on in-class evaluations is, of course, a violation of the Honor Code. http://www.cs.unc.edu/Admin/Courses/HonorCode.html

Prerequisites: Data structures and discrete mathematics, and the all-important mathematical maturity. You are expected to be able handle the material of CLRS ch. 3.2, 10.1-2, and app. B with brief review.

Courtesy: Remember that your assignment is to communicate with a human – be neat, complete, and concise. Please staple multi-page assignments. No cell phones ringing in class.

Suggestions: Come to class ready to think, and to make the instructor think.
Use lots of scratch paper. Collaborate – two heads can be better than one.
Look over assignments early so you brain has time to percolate on them.
Write your own versions of algorithms or proofs to make them your own.
Doing so gives you an early start on your cheat sheets, too.

Approximate Schedule:

January: Introduction to analysis of algorithms via sorting and searching. CLRS 1-5,7,AB
- Review relevant mathematics: invariants, induction, exponentials/logs, sums
- Asymptotic notation
- Algorithm design techniques: iterative and divide and conquer
- Algorithms: Binary search, Insertionsort, Mergesort, Quicksort
- Recurrence relations I
- Discrete probability and randomized algorithms

February: A little sorting and more searching. CLRS 6-7, 10-14, 4.
- Algorithms: Heapsort, Radix & Counting sort
- Recurrence relations II
- Data structures: Heap & priority queue, binary search trees, hash tables, red/black trees

March: Algorithm design techniques; Graphs. CLRS 15-17,22.
- Algorithm design techniques: Greedy algorithms, Dynamic programming, Amortization
- Data structures: Graphs (directed and undirected)

April: Graph algorithms CLRS 22-26, 34.
- Algorithms: BFS, DFS, Shortest paths, Network flow,
- NP-hard problems

Holidays:

Travel for Jack: Feb 2,4; Vishal Verma and Dave Millman will lecture.

Welcome to COMP 550, Algorithms and Analysis.

- **COMP 550 – Design and Analysis of Algorithms**
- **Spring 2010**
- **TTh 11:00-12:15, SN 007**
- **http://blackboard.unc.edu**
-
- Official:
    - **MATH 81 / OR 41** – Discrete Mathematics.
    - **COMP 121** – Data Structures.
- Specifically, I'll assume that you know, or can recall with a quick review, these chapters of CLRS:
    - Section 3.2: growth of functions
    - Chapter 10: elementary data structures
- Desirable:
    - **Elementary Probability** – We will introduce the basics of counting with probabilities, found in appendix C.

Course Roadmap

- Algorithmics Basics (2)
- Divide and Conquer (3)
- Randomized Algorithms (3)
- Sorting and Selection (6)
- Search Trees (3)
- Graph Algorithms (4.5)

- Greedy Algorithms (4)
- Dynamic Programming (2)
- Special Topics (1)
- Algorithmics Basics (2)
- Introduction to algorithms, complexity, and proof of correctness. (Chapters 1 & 2)
- Asymptotic Notation. (Chapter 3.1)
- ***Goals***
    - *Know how to write formal problem specifications.*
    - *Know about computational models.*
    - *Know how to measure the efficiency of an algorithm.*
    - *Know the difference between upper and lower bounds and what they convey.*
    - *Be able to prove algorithms correct and establish computational complexity.*
- Divide-and-Conquer (3)
- Designing Algorithms. (Chapter 2.3)
- Recurrences. (Chapter 4)
- Quicksort. (Chapter 7)
- ***Goals***
    - *Know when the divide-and-conquer paradigm is an appropriate one.*
    - *Know the general structure of such algorithms.*
    - *Express their complexity using recurrence relations.*
    - *Determine the complexity using techniques for solving recurrences.*
    - *Memorize the common-case solutions for recurrence relations.*
    - Randomized Algorithms (3)
- Probability & Combinatorics. (Chapter 5)
- Quicksort. (Chapter 7)
- Hash Tables. (Chapter 11)

- ***Goals***
    - *Be thorough with basic probability theory and counting theory.*
    - *Be able to apply the theory of probability to the following.*
        - *Design and analysis of randomized algorithms and data structures.*
        - *Average-case analysis of deterministic algorithms.*
    - *Understand the difference between average-case and worst-case runtime, esp. in sorting and hashing.*
- Sorting & Selection (6)
- Heapsort (Chapter 6)
- Quicksort (Chapter 7)
- Bucket Sort, Radix Sort, etc. (Chapter 8)
- Selection (Chapter 9)
- Other Sorting Methods (Handout)
- ***Goals***
    - *Know the performance characteristics of each sorting algorithm, when they can be used, and practical coding issues.*
    - *Know the applications of binary heaps.*
    - *Know why sorting is important.*
    - *Know why linear-time median finding is useful.*
- Search Trees (3)
- Binary Search Trees – Not balanced (Chapter 12)
- Red-Black Trees – Balanced (Chapter 13)
- ***Goals***
    - *Know the characteristics of the trees.*
    - *Know the capabilities and limitations of simple binary search trees.*
    - *Know why balancing heights is important.*
    - *Know the fundamental ideas behind maintaining balance during insertions and deletions.*

- *Be able to apply these ideas to other balanced tree data structures.*
- Graph Algorithms (3)

- Basic Graph Algorithms (Chapter 22)
- ***Goals***
  - *Know how to represent graphs* (*adjacency matrix and edge-list representations*).
  - *Know the basic techniques for graph searching.*
  - *Be able to devise other algorithms based on graph-searching algorithms.*
  - *Be able to "cut-and-paste" proof techniques as seen in the basic algorithms.*
  - Greedy Algorithms (4)
- Greedy Algorithms (Chapter 16)
- Minimum Spanning Trees (Chapter 23)
- Shortest Paths (Chapter 24)
- ***Goals***
  - *Know when to apply greedy algorithms and their characteristics.*
  - *Be able to prove the correctness of a greedy algorithm in solving an optimization problem.*
  - *Understand where minimum spanning trees and shortest path computations arise in practice.*
- Dynamic Programming (2)
- Dynamic Programming (Chapter 15)
- ***Goals***
  - *Know when to apply dynamic programming and how it differs from divide and conquer*.
  - *Be able to systematically move from one to the other*.
- Special Topics (1)
- Case Studies of Real-World Problems (lecture notes & handouts)
  - Algorithms in networking, operating systems, etc.

Goals

See how core algorithms can be put to use in real-world applications.

- Evaluation & Grading
- Weekly Homework & Quizzes: 30%
  (about 10, lowest score dropped)
    - Homework due Tues; some have Thurs "10-min quizzes"
- Two Midterm Exams: 40%
    - Weds: Feb 18, Mar 24
- Final Exam: 30%
    - Noon, Apr 26, sn 014.
- Bonus for Class Participation up to 5%.

Homework Assignments

- Can discuss in groups, but each student must turn in his or her own written solution.
- Please, be neat, clear, precise, formal.
    - Grading on correctness, simplicity, elegance & clarity.
- Communication
- Schedule, lecture notes, assignments, and gradebook are on UNC's blackboard site; log in with your onyen.

http://blackboard.unc.edu

Basic Courtesy

- Contribute to, but don't distract from class lectures and discussions.
- If you must arrive late or leave early, please take an aisle seat quietly.
- Write your assignments neatly, legibly, & formally.
- Use a stapler if you have several pages.

How to Succeed in this Course

- Start early on *all* assignments. *DON'T* procrastinate.
- Complete all reading before class.
- Participate in class.

- Think in class.
- Review after each class.
- Be formal and precise on all problem sets and in-class exams.