CSE 2320-001: Algorithms & Data Structures

Spring 2015: TR 3:30 - 4:50 p.m., Nedderman Hall 110

Instructor: Bob Weems, Associate Professor
Office: 627 ERB (`weems@uta.edu`, `http://ranger.uta.edu/~weems`)
Hours: T 11:15 a.m -1:15 p.m., R 1:00 - 3:00 p.m.

GTA: Contact information will be on my personal webpage

Prerequisites: C programming (CSE 1320, including basic UNIX competence)
Discrete Structures (CSE 2315, including combinatorics, trees, and graphs)

Objectives: In future design situations, students will be capable of developing, applying, and evaluating algorithmic solutions.

Outcomes:

1. Understanding of classic approaches to algorithm design - decomposition, dynamic programming, and greedy methods.
2. Understanding of particular algorithms and data structures that have wide applicabilty.
3. Understanding of basic algorithm analysis concepts by applying math skills to worst-case and expected time using recurrences and asymptotic notation.
4. Improved programming skills - especially data structures, recursion, and graphs.

Textbook: R. Sedgewick, *Algorithms in C*, *Parts 1-5, 3rd ed.*, Addison-Wesley, 2003.

References: S. Baase and A. Van Gelder, Computer Algorithms: *Introduction to Design and Analysis, 3rd ed.*, Addison-Wesley, 2000.

Cormen, Leiserson, Rivest, Stein, *Introduction to Algorithms, 3rd ed.,* MIT Press, 2009.

Readings: Indicated on calendar later in syllabus.

Homeworks: Six homeworks, with answers, are available on the course web page.

Grade: Based on the following weights:

Exams: 80% divided evenly among 3 exams.
Exam 3: Thursday, May 14, 2:00 - 4:30 p.m.

Programs: 20% divided among four assignments: 4%, 6%, 6%, 4%

Policies:

1. Regular attendance is expected. The lectures are being recorded and will have a link on the web page, but no availability guarantee is made (e.g. this is not a "distance" course).

2. Lecture notes and sample code for various algorithms are on the course web page `http://ranger.uta.edu/~weems/NOTES2320/cse2320.html`.

3. You are expected to have read the assigned readings by the specified date. Lectures will review and augment the material, but will also consider exercises from the book.

4. CHEATING - YOU ARE EXPECTED TO KNOW UNIVERSITY POLICIES. If you are suspected of cheating, the matter must go through university channels outside of the CSE Department. `http://www.uta.edu/conduct/`

5. Any request for special consideration must be appropriately documented <u>in advance</u>. (Special consideration does not include giving a higher grade than has been earned.)

6. Late programs are penalized according to the following schedule. LABS ARE DUE AT 3:15 PM ON THE DUE DATE, NOT MIDNIGHT. After the due time, assistance will not be provided.

Degree of lateness	Penalty
Up to 3:15 next day	5 pts
Up to 3:15 two days	15 pts
Up to 3:15 three days	30 pts
Up to 3:15 four days	60 pts

7. Each lab is graded as follows:

			Some Issues
a.	Output/Code	60%	If you know that your program has problems, you should let the GTA know what parts are functional. Test cases that demonstrate the limited functionality are useful.
b.	Internal Comments	6%	Beginning of file including `main()` should identify the assignment and who you are, along with giving a high-level description. Each function: identify each argument, describe processing, and each `return`. You may reference notes and text. Excess line-by-line comments are not needed, but the processing for each iteration of a (significant) loop should be explained.
c.	Modularity	6%	Functions are used appropriately. `main()` is kept simple.
d.	Structure	6%	Code is not unnecessarily complicated or long. It is often better to rewrite code rather than patching several times.
e.	Names	6%	Should indicate the purpose of the function, variable/field, or type. Cute or misleading names will be penalized.

f.	Spacing	6%	Indenting, blank lines, placement of {}. Be consistent.
g.	Generality	10%	Program is not unnecessarily limited.

All programs <u>must</u> be written in standard ***<u>C</u>*** to compile and execute on `omega.uta.edu`. Execution on other platforms (e.g. Visual Studio) does not assure compliance.

You are responsible for correctly submitting each programming assignment on Blackboard.

No points will be awarded for programs that do not compile. *Points for b-g will not be awarded to submissions that are not substantially complete and perform **<u>significant</u>** processing.* Submissions not reflecting algorithmic problem-solving techniques will not receive credit.

8. If you require a reasonable accomodation for a disability, please contact me no later than the second week of this semester. Further details are available at `http://www.uta.edu/disability`.

9. Occasional class-wide email messages (e.g. weather situations, clarifications) may be sent to the addresses recorded by MyMav. Messages will also be archived on the course web page.

Course Content (in chronological order)

1. Algorithmic Concepts (1.1-1.3, 6.1-6.3, 5.2, 8.1-8.7, 2.6, 12.4) - Disjoint Subsets, Selection Sort, Insertion Sort, Divide and Conquer, Mergesort (trivial recursion tree), Binary Search (with and without duplicates)
2. Growth of Functions (2.1-2.4, 2.6-2.7) - Asymptotic Notation (O, Ω, Θ), Upper Bounds, Lower Bounds
3. Summations - Geometric Series, Harmonic Series, Math Induction, Integrals
4. Recurrences (2.5) - Substitution Method, General Recursion Trees
5. Heapsort/Priority Queues (9.1-9.6) - Properties, Building a Heap, Sorting, Integrating with Other Data Structures
6. Greedy Algorithms - Quality-of-Solution Issues, Unweighted Interval Scheduling, Knapsack, Huffman Codes
7. Dynamic Programming (5.3) - Weighted Interval Scheduling, Optimal Matrix Multiplication, Longest Common Subsequence, Longest Increasing Subsequence, Subset Sum, Knapsack/Memoization

Exam 1: Items 1.-7.?.

8. Quicksort (7.1-7.8) - PARTITION (2 versions), Selection/Ranking
   Lower Bounds - Decision Tree Model, Stability (6.1)
   Counting (6.10) and Radix Sorts (10.1, 10.5)
9. Linked Lists (3.3, 2.6, 12.3, 3.5, 3.4) - Use in Dictionaries, Headers, Sentinels, Circular Lists, Double Linking
10. Stacks/Queues (4.2, 4.4, 18.1, 4.3, 4.6) - Policies and Applications
11. Rooted Trees (5.4-5.7) - Structure, Traversals
    Binary Search Trees (12.5-12.9) - Properties, Operations
12. Balanced Binary Search Trees (13.3-13.4) - Structural Properties, Rotations, Insertions

Exam 2: Items 7.?.-12.

13. Hashing (14.1-14.4) - Concepts, Chaining, Open Addressing

14. Graph Representations (3.7, 17.3-17.4) - Adjacency Matrices, Adjacency Lists, Compressed Adjacency Lists
    Search - Breadth-First (5.8, 18.7), Depth-First (19.2, 5.8, 18.2-18.4)
    Search-Based Algorithms - Topological Sort (19.6), Strong Components (19.8)
15. Minimum Spanning Trees (20.1-20.4) - Three Versions of Prim's MST, Kruskal's MST
16. Shortest Paths - Dijkstra's Algorithm (21.1-21.2), Warshall's Algorithm (19.3), Floyd-Warshall Algorithm (21.3)
17. Network Flows and Bipartite Matching (22.1, 22.2, 22.4) - Concepts, Augmenting Paths, Residual Network, Cuts, Max-flow Min-cut Theorem, Implementation, Performance Issues

Exam 3: Items 13.-17.

Calendar - with subject numbers from course content

January

20	Syllabus	22	1.
27		29	2.

February

3	3.	5	4.
10		12	5.
17	6.	19	7.
24		26	Exam 1

March

3		5	8.
10	SPRING	12	BREAK
17		19	9.
24	10.	26	
31	11.		

April

		2	12.
7	13.	9	14.
14	Exam 2	16	
21		23	15.
28	16.	30	17.

May

5		7	
		14	Exam 3

April 3 is the last day to drop; submit requests to major advisor prior to 4:00 p.m.