CISC 621 Algorithms, Final exam guide
(Exam is in 104 Gore Hall, 8am Tuesday, May 24)

The emphasis will be on the second half of the course. First half topics will account for at most one quarter of the final.

The final exam will have 5 parts.

Part 1: Some multiple choice and/or short answer questions.

For an example:
log* $(x) = 0$, if $x = 1$.
log* $(x) = 1 +$ log* (lg(x), if $x > 1$.
Which is true?

(a) log* (16) = 3.
(b) There are many more integers n for which log* (n) = 5 than there are for which log* (n) = 4.
(c) If log* (n) = 6, then n is a lot larger than the gross domestic product of the USA (in dollars).
(d) all of the above.

Part 2: [a correctness argument] Select one of the following two algorithms and explain why it correctly solves the corresponding algorithmic problem P, [a choice among 2 would be offered. See the indices of problems and algorithms in the course notes for the overall list from which the choices will be taken.] Each of the two algorithms and corresponding problems would be explicitly stated on the exam paper - you then explain why it works.]

Part 3: [a runtime analysis] Select one of the following two algorithms. State and explain the worst case runtime of the algorithm on inputs of size n. [list of two algorithms]

Part 4: [Data structure explanation] We have studied priority queues (binary heap and binomial heap implementations), hash tables, red-black trees, and dynamic disjoint sets (union-find). Be prepared to explain about structure and costs of one of these (most likely, it'll be hash table).

Part 5: Another question like one of parts 2, 3, or 4.

Explanations will be graded for clarity and thoroughness as well as correctness.

List of Topics: Problems and algorithms since Midterm

- Hash tables and hash functions. Chapter 11
  - Hash table, handling of hash collision
    - separate chaining
    - linear probing.
  - Hash functions
    - Simple uniform hash functions
    - Universal hash function set
- What is the parallel prefix operation?
- Network flows. Ford-Fulkerson algorithm. Reading: Chapter 26, sections 1,2,3. (augmenting path, residual graph, max flow = min cut)
- P, NP, NPC, and NP-Hard. Chapter 34
  - Definition of problem classes P, NP, NPC. Subset relations among them
  - Cook's theorem: Circuit-SAT is in NPC.
  - Polynomial time reductions
    - Circuit SAT reduces to (formula-)SAT (which reduces to 3-SAT)
    - 3-SAT reduces to Clique,
    - 3-SAT reduces to Subset-Sum.
    - Subset-Sum reduces to Partition
    - hwV reductions
- Graph Algorithms, Chapter 22
  - Graph types: directed or not, weighted or not, acyclic or not (DAGs).
  - Graph representation: matrix and array of lists.
  - Breadth first search, application to unweighted single source shortest path
  - Depth first search, application to topological sort and strong components.
- Minimal Spanning Trees, Chapter 23
  - Prim's algorithm
  - Kruskal's algorithm
  - Safe edge concept
- Dijkstra's single source shortest path algorithm, Chapter 24.3
- Dynamic Programming. Chapter 15
  - Memoization and option remember in Maple.
  - Top down design, followed by memoization or bottom up implementation to reduce memory.
  - LCS: Longest common subsequence
  - LIS: Longest increasing subsequence
  - Matrix chain product
- Fast polynomial and integer arithmetic via FFT, Chapter 30
  - Can be modular (3-primes FFT from Johnson's slides).
  - Basic: divide and conquer using $f(a) = fe(a^2) + a * fo(a^2), f(-1) = fe(a^2) - a * fo(a^2)$
  - n*log(n)

- Modular arithmetic for encryption and for hashing, Chapter 31
  In particular: RSA public key encryption. Chapter 31.7
  - Encryption and decryption using public and private keys uses modular exponentiation
  - Modular exponentiation is fast
  - Key construction involves
    * primality testing
    * Euler totient function: $\phi(n)$ is number of relatively prime $a < n$.
    * Euler's theorem: $a^{\phi(n)} = 1 \bmod n$.
    * Chinese remainder theorem
    * Extended Euclidean algorithm
- Blockchain (Bitcoin) is application of hashing and public key encryption

First half of course problem list

- Problem Max-min(A,n): Find both the maximum and minimum of an (unordered) array of n numbers. Algorithm in about 3n/2 comparisons, lower bound proof.
- Problem median(A,n): Find the element in an (unordered) array of n numbers that would be in the $\lfloor (n/2) \rfloor$ position if the array were sorted. Algorithm: use select.
- Problem select(A,n,k): Find the element of rank k in unordered array A of length n. [side issue: know meaning of *rank* in this context.] Algorithms: expected linear time `Randomized-Select` and worst case linear time `Select`,
- Problem sort: Algorithms insertionSort, heapSort, mergeSort, quickSort, introspectiveSort [Ch 2,6,7]
- Problem polynomial multiplication. Algorithm Karatsuba's divide and conquer.
- Problem matrix multiplication. Algorithm Strassen's divide and conquer. [Ch 4.2]
- Data Structure (Min)-Priority Queue (insert, extractMin). Implementation: binary heap, binomial heap. [Ch 6, exercise in Ch 19]
- Data Structure Map (insert, delete, search(=find) Implementation: hash table (issues:separate chaining or direct addressing, hash functions) [Ch 11]
- Data Structure Dictionary (insert, delete, search(=find), max, min, prev, next) Implementation: Left Leaning Red-Black trees [Ch 13, Sedgewick slides]

Tools: Master theorem [Ch 4], randomization [Ch 5]