MA/CSSE 473 Summer 2017 midterm exam specification (Last updated July 11, 2017)

Resources allowed:

1. Computer with webcam and microphone. You can use Moodle (but only to navigate to the Moodle Quiz that contains the exam, not to look things up), and a web page whose link I will give you that has the formulas from Appendix A.
2. In order to get the password that will allow you to view the Moodle quiz, you will have to connect *via* RemoteProctor.
3. An 8.5" x 11" sheet of paper, with handwritten notes on one side.
4. Calculator
    a. The only necessary operations should be +, *, -, /, exponentiation, and you probably won't even need those; you can probably do the problems in your head.
    b. You can also use a web page that does modular arithmetic and modular exponentiation. To get it, download and unzip the Calculator that I have provided at http://www.rose-hulman.edu/~anderson/Calc.zip
5. No other electronic devices, including phones, MP3 players, anything with headphones or earbuds.
6. I will give you the statement of the Master Theorem and the formulas from Appendix A.

Material that may be covered (obviously not all of it will be covered on a 2-hour test)**:**

- **Background material from 230:** Algorithms and analysis for implementing stacks, queues, linked lists, sequential lists, binary trees, binary search trees, binary heaps, dictionaries.
  Recursion and mathematical induction.
  Writing and solving simple recurrence relations, analysis of nested loops as in Weiss chapter 5 and its exercises.
  Applying the Master Theorem.
  Fibonacci numbers. Sequential and binary search.
  Well-known sorting methods: Insertion, selection, merge, quick, heap. Know algorithms and analysyis.
  Binary tree traversals: preorder, inorder, postorder, level order.
  Formal definitions of O(N), Θ(N), etc.
- **HW 1 - HW 8** (including the "not to turn in" problems)
- **Assigned readings for sessions 1-11** (Chapters 1-5 from the Levitin textbook as well as documents posted on Moodle that are listed on the Schedule page for Sessions 1-11),
  material from PowerPoint slides for sessions 1-17.
  There are more details below on specific algorithms that you should know.

Possible question types:

1. **T-F-IDK.** I will give you some statements. A statement is true (T) if it is always true. It is false (F) if there is at least one counterexample (i.e., sometimes false). You may also choose IDK to indicate that you do not know the answer. *There is some value (to me and you) in knowing what you don't know.*

Sample Point values: Correct answer: 5, incorrect answer: 0, IDK: 3, blank: 0. If you are totally guessing an answer, you are probably better off pointwise if you choose IDK. If you have a feeling that one "real" answer is correct, you are probably better off guessing that answer.

2. **Traditional multiple choice questions**
3. **Do you know the details of this algorithm?** Some questions will be checking to see whether you know and understand the details of specific algorithms. Here are examples of algorithms that you should be able to explain and to discuss the results of what happens when they are applied to specific input data.
   Examples (there are more examples in the background material section above)**:**
   (the first 6 bullets are form Dasgupta)
   - Addition, multiplication, exponentiation of integers (bit by bit, digit by digit), modular arithmetic.
   - Euclid's algorithm for finding GCD

- Extended Euclid for finding modular inverse, and finding a linear combination that leads to the GCD.
- RSA Cryptography encoding/decoding
- Fermat's little theorem. Uses and limitations when doing primality testing.
- Sieve of Eratosthenes (L 1.1)
- Towers of Hanoi (L 2.4)
- Sequential Search (L 3.2)
- Fast exponentiation (L 4.1)
- Primality testing: Fermat and Miller-Rabin. Know how and why they work.
- Generation of random prime numbers
- RSA Cryptography – how to encode and decode messages (need to be able to find modular inverses)
- DFS, BFS, topological sort
- Interpolation Search
- Permutation Generation:
- recursive minimal change,
- Johnson-Trotter,
- lexicographic
- Subset generation – Including Binary-reflected Gray Code
- Towers of Hanoi
- Closest Pair – divide-and-conquer algorithms
- QuickHull
- Strassen's matrix multiplication algorithm

4. **Concepts, definitions, etc**
   - Graph representations (adj matrix, adj lists) (L 1.4)
   - Graph definitions: edge, directed, head tail, digraph, loop, complete, dense, sparse, path, simple path, cycle, connected component (L 1.4)
   - Efficiency: best case, worst, average, amortized (L 2.1)
   - Asymptotic notation (O, Ω, Θ), including formal definition of big-O, limits and asymptotics (L 2.2)
   - Nested loops and summation (L 2.3, W Chapter 5)
   - Traveling Salesman problem, Knapsack Problem, Assignment problem (L 3.4)
   - Master Theorem for divide-and-conquer algorithms. No need to memorize; I will give you the formulas(L 5.1)

5. **Can you come up with a reasonably efficient algorithm to … ?** Many of the homework problems (especially the puzzle problems) have been this type of problem.

6. **Analyze the running time of some algorithm**
   May involve writing a recurrence relation and solving it.
   May involve counting nested loops by evaluating nested sums) and approximating with formulas from Appendix A)