# MA/CSSE 473 – Design and Analysis of Algorithms

## Homework 7 68 points total Updated for Summer, 2015

When a problem is given by number, it is from the textbook. 1.1.2 means “problem 2 from section 1.1” .

**Problems for enlightenment/practice/review (not to turn in, but you should think about them):**

How many of them you need to do serious work on depends on you and your background. I do not want to make everyone do one of them for the sake of the (possibly) few who need it. You can hopefully figure out which ones you need to do. These problems should all be review from CSSE 230.

5.1.1 [4.1.1] (divide-and-conquer array max for unsorted array)
5.1.2 [4.1.2] (divide-and-conquer array max/min for unsorted array)
5.1.7 [4.1.7] (Merge sort stability)
5.1.9 [4.1.9] (O(n log n) algorithm to count inversions in an array)
5.2.1 [4.2.1] (quicksort example)
5.2.4 [4.2.4] (quicksort sentinel)
5.2.6 [4.2.6] (increasing arrays in quicksort)

**Problems to write up and turn in:**

Problems 1-2 are based on the Dasgupta excerpt that is available on Moodle (in the Reading Materials folder). Most of the material is also covered in Weiss, Sections 7.4 and 9.6, which are also on Moodle.
**Note that my python code for Euclid, Extended Euclid, and modexp are linked from the schedule page. Feel free to use them.**

1. (15) (RSA decoding). If small primes are used, it is computationally easy to "crack" RSA codes. Suppose my public key is N=703, e= 53. You intercept an encrypted message intended for me, and the encrypted message is 361. What of RSA was the original message? How did you get your answer? [RSA details are found in Dasgupta, and Weiss section 7.4.4]

2. ( 6) (RSA attacks) Find and read about various ways of attacking the RSA cryptosystem. Write about two attacks that interest you. Explain how they work.

3. ( 3) 5.1.4 [4.1.4] (logarithm base in the Master Theorem)
4. ( 6) 5.1.5 [4.1.5] (Simple application of the Master Theorem)
5. ( 6) 5.2.2 [4.2.2] (Quicksort partition scan properties) Note that the old (2nd) edition of the Levitin book has a part c, and I want you to do it, you can find it in the http://www.rose-hulman.edu/class/csse/csse473/201540/Homework/hw07_levitin_probs.pdf document.
6. (10) Solve the average-case recurrence for quicksort. The recurrence is given on page 180 [133] of Levitin. Feel free to look up a solution, understand it, and write it in your own words (and symbols). The Weiss Data Structures book (Section 8.6.2) is one place that has a solution. You should write a reasonable amount of detail, enough to convince me that you understand it.
7. ( 6) 5.2.8 [4.2.8] (Negatives before positives)
8. ( 8) 5.2.9a [4.2.9] (Dutch National Flag) [do it with a one-pass algorithm if you can]
9. ( 8) 5.2.11 [4.2.11] (Nuts and bolts). In addition to writing the algorithm, write and try to solve a recurrence for average-case efficiency.