Problem Set 2 for CS 170

Problem 0

How much time did you spend on the last problem set? What about this one (do this "problem" last, of course)? Which problem did you spend the most time on? (And how much time was that?)

Please try to include all the time spent thinking about the problems, discussing them with your study group or at office hours, and writing up the solutions, but not time spent playing nethack with your laptop on top of this sheet. Feel free to add further comments about the difficulty level of the assignments thus far. As always, problem 0 will be graded.

Problem 1

Compute $13^{37^{170170}} \bmod 23$.

Problem 2

Problem 2 **will not count** for this homework. Any solutions submitted or lack thereof will not affect your grade on the problem set. Coming later: the solution set will feature what we *thought* was a reasonable proof for the original problem, and why it was wrong. The only other proof we know of at the moment involves some mathematics which are not covered in the pre-requisites. Here's the original problem, just for reference; once again, this problem **will not count** as part of the grade:

Let N be a Carmichael number. Prove that no prime occurs more than once in the prime factorization of N (that is, there is no prime p such that N/p is an integer and is still divisible by p).

Problem 3

In this problem, we will explore RSA and *digital signatures*. In a *digital signature scheme*, there are two algorithms, Sign and Verify. The algorithm Sign takes a secret key and a message, then outputs a signature σ. Verify takes a public key, a signature σ, and a message M, then returns "true" if σ could have been created by Sign with that key on message M and "false" otherwise.

(a) Why would we want digital signatures? (1 paragraph or less)

(b) An RSA digital signature consists of Sign(d,M)= $M^d \pmod n$, where d is an RSA private exponent. Prove that anyone who knows the corresponding public exponent e can perform Verify(e, M, M^d); i.e. they can check that the signature really was created by the private exponent.

(c) Generate your own RSA modulus $n = pq$, public exponent e, and private exponent d. You are not required to use a computer — you may use a small modulus and work by hand. Now sign a message $m \leq n$ of your choice. Show us the resulting signature, and how to verify it.

(d) Signing involves decryption, and is therefore risky. If Bob signs anything his is asked to, Eve can take advantage of this and decrypt a message $y = x^e \bmod n$ that has been sent by Alice to Bob. What message would Eve ask Bob to sign?

(e) Finally, suppose that Bob is more careful, and refuses to sign messages whose signatures look suspiciously like text. How could Eve still swindle him?

Problem 4

In this problem, and for the rest of the class, unless otherwise explicitly stated, you should assume that all integers involved are relatively "small", so all arithmetic operations (such as comparisons between 2 integers) take $O(1)$ time.

(a) Give a divide-and-conquer algorithm for determining the largest and second largest values in a given unordered set of n numbers which runs in $\Theta(n)$ time.

(b) Can we use the same approach without significant modifications to find the third-largest value in the set in linear time? Why or why not?

(c) Can we use the same approach without significant modifications to find the median of the set (the $\frac{n}{2}$'th-largest value) in the set in linear time? Why or why not?

Problem 5

Give asymptotically tight solutions for $T(n)$. (In other words, find an explicit formula $f(n)$ so that $T(n) = \Theta(f(n))$, and prove it.)

(a) $T(n) = 2T(n/4) + \sqrt{n}$

(b) $T(n) = T(n-d) + T(d) + c \cdot 2^n$, given constants $d \geq 1$, $c > 0$. Use a recursion tree to guess a solution, then prove your guess by induction. Assume $T(n) = \Theta(1)$ for $n \leq d$.

(c) $T(n) = 3T(\sqrt[3]{n}) + c$, for a constant c.