Design and Analysis of Algorithms — Neal Young
Computer Science 45
Midterm Exam, due in class Monday, May 12, 1997

Do **three** of the following problems.[1] You can consult your notes, the text, or me when working on this exam, but please don't consult any other source. If you get stuck on a problem, you can come to me and "buy" a hint at the expense of part of the credit for the problem.

1. **Two-dimensional pattern matching:**

   Design and analyze an algorithm for the following problem. Your algorithm may be randomized. It should be as efficient as you can make it, in the "big-O" sense. Give as complete an analysis as you can. ("Complete" does not mean overly detailed, it just means that there are no gaps or "hand waving".)

   Given: Two two-dimensional arrays $T[1..n, 1..n]$ and $P[1..m, 1..m]$, where $m \leq n$ and each array entry is a digit in $\{0, 1, \ldots, 9\}$.

   Question: Does P occur in T? More specifically, are there an a and a b such that $T[i+a, j+b] = P[i, j]$ for all i and j between 1 and m?

2. **Riemann zeta function:**

   This problem asks you to consider an example of a different style of generating function.

   Define $\zeta(z) = \sum_{n=1}^{\infty} \frac{1}{n^z}$.

   (a) Define d_n so that $\zeta(z)^2 = \sum_{n=1}^{\infty} \frac{d_n}{n^z}$. Express the quantity d_n in words.

   (b) Argue that
   $$\zeta(z) = \prod_p \frac{1}{1 - p^{-z}}$$
   where p ranges over all the primes.[2]

   Why does this imply that there are infinitely many primes?[3]

3. **Lift-to-front algorithm:**

   Read section 27.5 of CLR to understand the lift-to-front algorithm, then give a different analysis of the algorithm. Your analysis should be an amortized analysis based on a *potential function.* (The challenge here is to design the potential function.)

4. **Integer factorization:**

   Professor Bo Zo wanted to assign an exam question on the Pollard-Rho algorithm, but he couldn't understand it. Read section 33.9, to understand Pollard-Rho, then help clear up Bo's questions:

   (a) Instead of using the function $x_{i+1} = x_i^2 - 1 \bmod n$, wouldn't any function do, as long as it "looks random"? For each of the following functions suggested by Bo, explain whether that function will do as well, and if not what is wrong with it.

      i. $x_{i+1} = 17x_i \bmod n$
      ii. $x_{i+1} = 2^{x_i} - 1 \bmod n$
      iii. $x_{i+1} = x_i^3 - 1 \bmod n$
      iv. $x_{i+1} = ix_i - 1 \bmod n$

   (b) Does the argument given really use the assumption (made in the last paragraph of page 846) that $\gcd(p, n/p) = 1$?[4]

   If it *doesn't* use that assumption, then can't the analysis be improved, because the time to find a factor will be $O(\sqrt{p_1})$, rather than $O(\sqrt{p_1^{e_1}})$ (an improvement when $e_1 \neq 1$)?

   If it *does* use that assumption, then how does the algorithm work when applied to a prime power (i.e., $n = p_1^{e_1}$ with $e_1 > 1$)? If it doesn't work, how do you handle this case?

---

[1] Do all four for extra credit, but make sure you indicate which one you want to count as "extra".
[2] Recall that "$\prod$" denotes a product just as "$\sum$" denotes a sum.
[3] Recall that the series $\sum_n 1/n$ diverges.
[4] At first Bo thought that (33.50) required the assumption for the Chinese remainder theorem to apply, but then he decided that (33.50) would hold for *any* p that divided n.