CS 70 Discrete Mathematics for CS
Spring 2005 Clancy/Wagner

HW 8

Due Thursday, March 31

Coverage: This assignment involves topics from the lectures of March 1, 8, 10, 15, 17, and from Rosen section 2.6 and chapter 4.

Administrative reminders: We will accept only unformatted text files or PDF files for homework submission. Include your name, login name, section number, and partner list in your submission. Give the command `submit hw8` to submit your solution to this assignment.

Homework exercises:

1. **(6 pts.) Repeated squaring**
   The algorithm for computing $a^b \bmod c$ by repeated squaring does not necessarily lead to the minimum number of multiplications. Give an example of b ($b > 10$) where the exponentiation can be performed using fewer multiplications, by some other method.

2. **(10 pts.) RSA**
   Let p and q be primes and let $N = pq$. Show how to determine p and q given N and $(p-1)(q-1)$. (In other words, given the public key (e,N), e the encryption exponent and N the RSA modulus, and the value $\varphi(N) = (p-1)(q-1)$, it is possible to compute p and q by simple (polynomial time) algebraic operations. This shows that determining $\varphi(N)$ is "as hard as factoring.")

3. **(12 pts.) Secret sharing**
   (a) Suppose that the teaching staff of a course involves three professors and two TAs. The solutions to the next homework are encrypted by an encryption key shared by all five. The three professors together should be able to access the solutions, or any one TA with one professor, or both TAs. Suggest a secret-sharing scheme that achieves this. (*Hint:* Try weights.)
   (b) Suppose now that the class is taught by three professors, each with her own two TAs. Any two professors can access the data, as long as one of each professor's TAs (i.e. a total of at least four people) is also present. Now what?

4. **(20 pts.) Polynomial interpolation**
   (a) Prove the following: If p is a prime and $y_1, \ldots, y_n \in \mathbf{N}$ are all different from 0 modulo p, then $y_1 \times \cdots \times y_n$ is also different from 0 modulo p.
   (b) Prove the following: Given a prime p and two integers a, b, it is always possible to find a polynomial $f(x)$ of degree at most 1 such that $f(0) \equiv a \pmod{p}$ and $f(1) \equiv b \pmod{p}$.

(c) You are given a prime p and a positive number $n < p$. Show how to find a polynomial $f(x)$ of degree at most n satisfying $f(0) \equiv f(1) \equiv \cdots \equiv f(n-1) \equiv 0 \pmod{p}$ and $f(n) \equiv 1 \pmod{p}$. In other words, the polynomial f should be congruent to zero at the points $x = 0, \ldots, n-1$; at $x = n$ the polynomial should be 1 mod p.

Hint: Consider $F(x) = (x-0)(x-1)(x-2)\cdots(x-(n-1))$; what can you say about it?

(d) You are given p and n as before, but now you are also given an index j with $0 \le j \le n$. Show how to find a polynomial $g_j(x)$ of degree at most n satisfying

$$g_j(i) \equiv \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} \pmod{p} \qquad\qquad \text{for each } i = 0, 1, \ldots, n.$$

In other words, the polynomial g_j should be congruent to zero at the points $x = 0, \ldots, n$, except that at $x = j$ it should be congruent to 1 mod p.

(e) You are given a prime p, a number n with $0 < n < p$, and a sequence of values $a_0, a_1, \ldots, a_n \pmod{p}$. Describe an efficient algorithm to find a polynomial $h(x)$ of degree at most n satisfying $h(0) \equiv a_0 \pmod{p}$, $h(1) \equiv a_1 \pmod{p}$, $\ldots$, $h(n) \equiv a_n \pmod{p}$.

Hint: What can you say about the polynomial $3g_0(x) + 7g_1(x)$, where $g_0(x), g_1(x)$ are as defined in part (d)? Does this give you any ideas?

5. (12 pts.) Bit string counting

How many bit strings of length 10 contain either (a) at least five consecutive 0's or (b) at least five consecutive 1's? Show your work.