

## Homework 4

*Instructor: Anna Lysyanskaya**Due: February 21, 2007*

For a more in-depth treatment of this material, see Victor Shoup's book [A Computational Introduction to Number Theory and Algebra](http://shoup.net/ntb/), freely available for downloading at <http://shoup.net/ntb/>.

The following notation is useful for this homework:

**Definition 1.** Let  $n$  be an integer.  $\mathbb{Z}_n^*$  denotes the set of integers such that  $x \in \mathbb{Z}_n^*$  iff  $1 \leq x \leq n$  and  $\gcd(x, n) = 1$ .

## Problem 1: Extended Euclidean GCD Algorithm

On input integers  $x$  and  $y$ , the extended Euclidean GCD algorithm finds integers  $a$  and  $b$  such that  $ax + by = \gcd(x, y)$ . This is done by careful bookkeeping throughout the execution of the Euclidean GCD algorithm, and is described in more detail in Shoup, Chapter 4.

- Use Euclid's algorithm to compute  $\gcd(254, 366)$
- Use extended Euclidean Algorithm to find  $L, K$  such that  $16L + 35K = 1$ .
- Find  $16^{-1} \pmod{35}$ .
- Find  $84^{-1} \pmod{143}$ .

## Problem 2: Practice with Chinese Remainder Theorem

In class we saw a simplified version of the Chinese Remainder Theorem, where  $n$  was the product of 2 primes. But this can be extended to allow  $n = \prod_{i=1}^k n_i$  (as long as the  $n_i$  are relatively prime). For a review of the Chinese Remainder Theorem, see Section 2.2 in Shoup.

Let  $n = 11 * 13 * 16$ . Suppose that we know that :

$$x \equiv 8 \pmod{11}$$

$$x \equiv 2 \pmod{13}$$

$$x \equiv 7 \pmod{16}$$

Let's use the Chinese remainder theorem to find  $x \pmod{n}$ .

- Use extended Euclid's algorithm to find  $a, b$  such that  $1 = a * 11 + b * 13 * 16$ .
- Use extended Euclid's algorithm to find  $a, b$  such that  $1 = a * 13 + b * 11 * 16$ .
- Use extended Euclid's algorithm to find  $a, b$  such that  $1 = a * 16 + b * 11 * 13$ .
- Use CRT to find  $x \pmod{11 * 13 * 16}$ .

### Problem 3: Micali's Primality Test

**Lemma 1.** *Let  $n = pq$  be an RSA modulus. Every square  $a^2 \in \mathbb{Z}_n^*$  has exactly four square roots:  $(a \bmod p, a \bmod q)$ ,  $(a \bmod p, -a \bmod q)$ ,  $(-a \bmod p, a \bmod q)$ , and  $(-a \bmod p, -a \bmod q)$ .*

**Lemma 2** (Fermat's Little Theorem). *Let  $p$  be a prime, and  $a \in \mathbb{Z}_p^*$ . Then  $a^{p-1} \equiv 1 \pmod p$ .*

In this problem, we will look at Micali's primality test. Suppose  $SQRT(a, p)$  is a polynomial-time algorithm that, if  $p$  is prime and  $a \in \mathbb{Z}_p^*$  is a quadratic residue, outputs some square root of  $a \bmod p$ .

- a. Prove that the following algorithm is a primality test: on prime input it outputs PRIME with high probability and on composite input it outputs COMPOSITE with probability at least  $1/2$ .

Algorithm:

Input:  $n$  an integer

- If  $n = 2$ , output PRIME;
  - If  $n$  is even, output COMPOSITE;
  - If  $n$  is of the form  $n = p^b$  for some  $p, b > 1$ , output COMPOSITE;
  - Choose a random  $r \in \{1, \dots, n-1\}$ , If  $\gcd(r, n) > 1$ , output COMPOSITE;
  - $x \leftarrow SQRT(r^2 \bmod n, n)$ ; If  $x = \pm r \bmod n$ , output PRIME; Else output COMPOSITE;
- b. Suppose  $p = 4m + 3$  is a prime and that  $a$  is a quadratic residue modulo  $p$ . Prove that  $a^{m+1}$  is a square root of  $a$  modulo  $p$ .
- c. Use (a) and (b) to devise a primality testing algorithm for integers equal to 3 modulo 4.