Problem 1: The 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 (see the links page of the website).

Note that you need not include your work— but make sure to check your final answer!

a. Use the Euclidean algorithm to compute $\gcd(1053, 403)$.

b. Use the Extended Euclidean Algorithm to find $L, K$ such that $16L + 55K = 1$.

c. Find $16^{-1} \mod 55$.

d. Find $1245^{-1} \mod 143$.

Problem 2: Practice with the Chinese Remainder Theorem

Recall the Chinese Remainder Theorem, which we saw in class. (It is also in Section 2.4 of the Shoup textbook.)

Let $n = 11 \cdot 23 \cdot 31$. Suppose that we know that

$$x \equiv 4 \mod 11$$
$$x \equiv 7 \mod 23$$
$$x \equiv 9 \mod 31$$

Let’s use the Chinese remainder theorem to find $x \mod n$.

a. Use the extended Euclidean algorithm to find $a, b$ such that $1 = a \cdot 11 + b \cdot 23 \cdot 31$.

b. Use the extended Euclidean algorithm to find $a, b$ such that $1 = a \cdot 23 + b \cdot 11 \cdot 31$.

c. Use the extended Euclidean algorithm to find $a, b$ such that $1 = a \cdot 31 + b \cdot 11 \cdot 23$.

d. Use the CRT to find $x \mod 11 \cdot 23 \cdot 31$.

Problem 3: Micali’s Primality Test

**Lemma 1.** Let $n = pq$ be a product of two relatively prime odd integers $p$ and $q$. Every square $a^2 \in \mathbb{Z}_n^*$ has at least four square roots: $(a \mod p, a \mod q)$, $(a \mod p, -a \mod q)$, $(-a \mod p, a \mod q)$, and $(-a \mod p, -a \mod q)$.

In this problem, we will review Micali’s primality test we saw in class. Suppose $\text{SQRT}(a, p)$ 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 $a \mod p$. 

HW 3-1
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. (Note that this probability is over the random choices that the algorithm makes, and should hold for all inputs to the algorithm.)

**Input:** n an integer

if \( n = 2 \) then output PRIME ;

if \( n \) is even then output COMPOSITE ;

if \( n \) is of the form \( n = p^b \) for some \( p, b > 1 \) then output COMPOSITE ;

Choose a random \( r \in \{1, \ldots, n-1\} \);

if \( \gcd(r, n) > 1 \) then output COMPOSITE ;

\( x \leftarrow \text{SQRT}(r^2 \mod n, n) \);

if \( x = \pm r \mod n \) then output PRIME else output COMPOSITE ;

**Algorithm 1:** Primality test

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 congruent to 3 modulo 4.

d. Prove the lemma stated in the beginning of this problem.

---

**Problem 4: Statistical Indistinguishability**

Suppose we have two possibly overlapping sets \( S_{0,k}, S_{1,k} \subseteq \{0,1\}^k \). Let \( D_{0,k} \) be the distribution created by choosing each element \( x \in S_{0,k} \) with probability \( P_{0,k}(x) \) and let \( D_{1,k} \) be the distribution created by choosing each element \( x \in S_{1,k} \) with probability \( P_{1,k}(x) \).

Recall our definition of statistical indistinguishability. We first define the advantage of an algorithm Test:

\[
\text{Adv}(Test) = |\Pr[b \leftarrow \{0,1\}; x \leftarrow D_{b,k}; b' \leftarrow Test(x) : b' = b] - 1/2|
\]

(This is slightly different than the way we defined it in class, in that here we insist that the advantage be a positive quantity.)

Two distributions \( D_{0,k} \) and \( D_{1,k} \) are statistically indistinguishable if for all algorithms Test, there exists a negligible function \( \nu(\cdot) \) such that \( \text{Adv}(Test) = \nu(k) \).

Intuitively, think of Test as an algorithm that gets an element \( x \) and tries to determine whether it was chosen from \( D_{0,k} \) or \( D_{1,k} \). If it thinks it came from \( D_{0,k} \) it will output 0, and if it thinks it came from \( D_{1,k} \) it will output 1. If it is just as likely (or within a negligible amount) to output 0 given an element chosen from the first distribution as it is given an element chosen from the second distribution, then that means it truly cannot tell the two distributions apart.

a. Suppose we have a distinguisher Dave who knows everything about both distributions. In particular, he knows the probability of every element of each distribution. What is Dave’s optimal strategy for distinguishing elements from these two sets (i.e. what is the best Test algorithm)?

b. Let \( \text{Adv}'(Test) = (|p_0(\text{Test}) - p_1(\text{Test})|)/2 \), where for \( b \in \{0,1\} \),

\( p_b(\text{Test}) = \Pr[x \leftarrow D_{b,k}; b' \leftarrow Test(x) : b' = b] \)

Show that if the only possible outputs of \( Test \) are 0 and 1, then \( \text{Adv}(Test) = \text{Adv}'(Test) \).
c. Now consider the following: we define the statistical distance between two sets as

\[
\Delta(D_{0,k}, D_{1,k}) = \frac{1}{2} \sum_{\hat{x} \in \mathcal{S}_1(k) \cup \mathcal{S}_0(k)} |\Pr[x \leftarrow D_{1,k} : x = \hat{x}] - \Pr[x \leftarrow D_{0,k} : x = \hat{x}]|.
\]

**Claim.** \(\Delta(D_{0,k}, D_{1,k}) = \nu(k)\) for \(\nu\) negligible if and only if \(D_{0,k}\) and \(D_{1,k}\) are statistically indistinguishable.

Prove this claim (Hint: use the fact that in part (a) we found the best possible Test algorithm and evaluate Adv or Adv', whichever you find more intuitive, of that algorithm).