Solution for PS7-1.

a. If $M = 0$, clearly $M' = 0$. Else, since $ed \equiv 1 \pmod{P-1}$, write $ed = k(P-1)+1$ where $k \in \mathbb{N}$. We compute

$$M' \equiv C^d \equiv M^{ed} = (M^{P-1})^k \cdot M \equiv 1^k M = M \pmod{P},$$

where the last congruence is because of Fermat's Little Theorem. Since $M, M' \in \mathbb{Z}_P$, $M = M'$.

b. Dr. Speedy's "cryptosystem" is not secure at all! Anyone can use the Extended GCD Algorithm to compute d from e and P (which are both public) and then cheerfully decrypt any message to Bob that they can intercept.

Solution for PS7-2. Suppose that $N = pq$, where p and q are distinct primes. Using the result of a previous class exercise, $\phi(N) = \phi(pq) = (p-1)(q-1) = N - p - q + 1$.

Therefore, $q = N - \phi(N) + 1 - p$. Substituting this expression for q in $N = pq$, we obtain

$$p(N - \phi(N) + 1 - p) = N,$$
$$\text{i.e.,} \quad p^2 + (\phi(N) - N - 1)p + N = 0.$$

Run algorithm $\mathscr{A}$ to obtain $\phi(N)$. We now know all the coefficients in this quadratic equation for p. Solving it, we obtain p. Then we obtain $q = N/p$.

Solution for PS7-3. One can verify that the Decryption Theorem for RSA still holds with 10-prime RSA, so Dr. Tricksy's idea is not immediately flawed.

For regular, 2-prime RSA, with public key (N_2, e_2), and private key d, the time needed to perform encryption and decryption is given by the time needed for modular exponentiation. This mainly depends on the number of bits needed to express N_2 and e_2. Given the value of N_2, the value of e_2 is essentially unrestrained; it must only be coprime to $\phi(N)$. For 10-prime RSA, and a modulus N_{10} with the same number of bits as N_2, because only a small fraction of numbers are not coprime to N_{10}, one can pick an encryption exponent e_10 almost exactly equal to e_2. As a result, for the same key size, encryption and decryption procedures do not differ significantly between 2-prime and 10-prime RSA.

On the other hand, for the same key size (number of bits of N_2 and N_{10}), the time to factor the modulus can differ significantly for for 10-prime RSA. Here we consider the brute force factoring algorithm, that checks divisibility by every number $\leq \sqrt{n}$. Better algorithms, like the General Number Field Sieve or the Elliptic-Curve Factorization Method, are more efficient, but their runtime analysis is more complicated. Write the factors of N_2 and N_{10} in increasing order, so that $N_2 = p_1 \cdot p_2$, and $N_{10} = q_1 \cdot q_2 \cdot q_3 \cdots q_{10}$, with $p_1 < p_2$, and $q_1 < q_2 \ldots < q_{10}$. Then brute force factorization of N_2 requires time $O(p_1)$ to discover the smaller factor; while factoring N_{10} requires time $O(q_1 + q_2 + \ldots + q_9) = O(q_9)$ to discover the 9 smallest factors.

To make breaking 2-prime RSA difficult, p_1 and p_2 are chosen roughly equal in size, so that $p_1 \approx \sqrt{N_2}$, and it takes $O(\sqrt{N_2})$ time to brute-force factor N_2. To make breaking 10-prime RSA hard, q_9 should be as large as possible. This can be done by setting $q_1 = 2$, $q_2 = 3$, $q_3 = 5$, and picking q_9 and q_{10} both $\approx \sqrt{N_{10}}$, in which case the time to factor N_{10} is the a constant multiple of that for N_2. There isn't any reason to prefer 10-prime RSA over 2-prime RSA.

If the prime factors of N_{10} are all roughly the same size, then $q_1 \approx q_2 \approx \ldots q_{10} \approx N_{10}^{1/10}$, and it only takes $O(N_{10}^{1/10})$ time to break Dr. Tricksy's method, which is far less than $O(\sqrt{N_2})$.

Solution for PS7-4.

a. When we call `modpow_bad(a, k, n)`, it results in k calls to `modmult`. Since k is a 1024-bit integer, the value of k could be as large as $2^{1024} - 1$ and is typically greater than 2^{1000}. Even if each call to `modmult` takes just one nanosecond, the time spent by `modpow_bad` would be orders of magnitude greater than the age of the universe!

b. By repeated squaring, compute $a \to a^2 \to a^4 \to a^8 \to a^{16} \to a^{32} \to a^{64}$ (all computations modulo n). Then combine the results like this: $a \cdot a^2 \cdot a^{16} \cdot a^{64} = a^{1+2+16+64} = a^{83}$ (again, modulo n).

c. If n is even, let $n = 2k$. Then $x^n = x^{2k} = (x^2)^k = (x^2)^{\lfloor n/2 \rfloor}$.
If n is odd, let $n = 2k+1$. Then $x^n = x^{2k+1} = x \cdot (x^2)^k = x \cdot (x^2)^{\lfloor n/2 \rfloor}$.

d. The equation in the previous part shows how raising to the power n can be reduced to raising to the power $\lfloor n/2 \rfloor$, provided we first compute the square. We can translate this directly into the following recursive implementation.

```
def modpow(a, k, n):
    """compute a**k modulo n quickly, assuming k >= 0, n > 0"""
    if k == 0:
        return 1
    square = modmult(a, a, n)
    temp = modpow(square, k // 2, n)
    if k % 2 == 0:
        return temp
    else:
        return modmult(a, temp, n)
```

e. When I ran the above code on the given numbers, it "instantly" returned 4808550559.

Solution for PS7-5.

a. Since $\gcd(m,n) = 1$, by ***PS5-6***$^{\textit{HW}}$, $\mathrm{lcm}(m,n) = mn$. Therefore, by ***PS6-9***, if $m \mid s$ and $n \mid s$, then $mn \mid s$. Now, to prove what's asked for, take $s = x - y$.

b. Suppose that $f(x) = f(y)$. Then $(x \bmod m, x \bmod n) = (y \bmod m, y \bmod n)$. In other words, $x \equiv y \pmod{m}$ and $x \equiv y \pmod{n}$. Therefore, by the previous part, $x \equiv y \pmod{mn}$. Since both x and y belong to $\mathbb{Z}_{mn}$, this forces $x = y$.

c. Since f is injective, $|\mathrm{Range}(f)| = |\mathrm{Domain}(f)| = |\mathbb{Z}_{mn}| = mn$. However, $|\mathrm{Codomain}(f)| = |\mathbb{Z}_m \times \mathbb{Z}_n| = mn$ as well. Therefore, $\mathrm{Range}(f) = \mathrm{Codomain}(f)$, which makes f surjective.

d. By the previous two parts, f is bijective. Therefore, for each $(a,b) \in \mathbb{Z}_m \times \mathbb{Z}_n$, there is one and only one value $x \in \mathbb{Z}_{mn}$ such that $f(x) = (a,b)$. This value, which is precisely $f^{-1}(a,b)$, is the unique solution to the system of congruences.

Solution for PS7-6.

a. We already know from ***PS7-5*** that f is injective on the full domain $\mathbb{Z}_{mn}$. Therefore, we only need to show that $x \in \mathbb{Z}_{mn}^* \iff f(x) \in \mathbb{Z}_m^* \times \mathbb{Z}_n^*$. This is logically equivalent to $x \notin \mathbb{Z}_{mn}^* \iff f(x) \notin \mathbb{Z}_m^* \times \mathbb{Z}_n^*$, so we'll prove the latter instead.

Suppose that $x \notin \mathbb{Z}_{mn}^*$. Then $\gcd(x, mn) = d > 1$. Let p be a prime divisor of d. Then $p \mid x$ and $p \mid mn$. By Euclid's Lemma, either $p \mid m$ or $p \mid n$. Assume WLOG that $p \mid m$. Let $f(x) = (a,b)$. Then

$$\gcd(a,m) = \gcd(x \bmod m, m) = \gcd(x,m) \geq p > 1\,,$$

whence $a \notin \mathbb{Z}_m^*$, implying $f(x) \notin \mathbb{Z}_m^* \times \mathbb{Z}_n^*$.

On the other hand, suppose that $f(x) = (a,b) \notin \mathbb{Z}_m^* \times \mathbb{Z}_n^*$. Assume WLOG that $a \notin \mathbb{Z}_m^*$. Then $\gcd(a,m) = d > 1$. Therefore,

$$\gcd(x, mn) \geq \gcd(x,m) = \gcd(x \bmod m, m) = \gcd(a,m) = d > 1\,,$$

whence $x \notin \mathbb{Z}_{mn}^*$.

b. The existence of a bijection from finite set A to finite set B implies $|A| = |B|$. Therefore, $\phi(mn) = \phi(m)\phi(n)$. In the language of Unit 8, we are using the bijection principle and the product principle.

Solutions for Unit 7

Prof. Amit Chakrabarti
Computer Science Dept
Dartmouth College

Solution for PS7-7.

a. A square root of unity modulo m is exactly the same thing as a self-inverse modulo m. If m were prime, as shown in ***PS5-9*** ^{HW}, it would have had only two such square roots: 1 and $m-1$.

b. By definition, $b^2 \equiv 1 \pmod{m}$, so $m \mid b^2 - 1 = (b-1)(b+1)$. Let p be a prime divisor of m. Then $p \mid (b-1)(b+1)$, so by Euclid's Lemma, either $p \mid b-1$ or $p \mid b+1$. Since $b \neq 1$, $b \neq m-1$, and $m \geq 3$, we have $1 \leq b-1 < b+1 < m$.

Suppose that $p \mid b-1$. Then $\gcd(m, b-1) \geq p > 1$. Also, $\gcd(m, b-1) \leq b-1 < m$. Therefore $\gcd(m, b-1)$ is a nontrivial divisor of m.

Suppose that $p \mid b+1$. Then $\gcd(m, b+1) \geq p > 1$. Also, $\gcd(m, b+1) \leq b+1 < m$. Therefore $\gcd(m, b+1)$ is a nontrivial divisor of m.