Name: ________________________________

- You may not use any reference materials during this exam.
- Electronic devices, including calculators, cell phones, mp3 players, and laptops are all prohibited.
- You may not use your own scratch paper. The exam has plenty and you can ask for more if needed.
- You may not leave the classroom once the exam has been distributed.
- Communicating with other students in any way is prohibited.

Academic Honesty: I understand that if I cheat on this exam in any way, I will receive the maximum possible penalty, including an F in this course.

Name (print): ________________________________

Signature: ________________________________
Problem 1

Modular Arithmetic Compute the following:

Solutions in-line

• (5 points) $\text{gcd}(51, 78)$. The extended Euclidean algorithm computes:

\[
\begin{array}{c}
[1, 0, 78] \\
[0, 1, 51] \quad k = 1 \\
[1, -1, 27] \quad k = 1 \\
[1, -1, 27] \quad k = 1 \\
[2, -3, 3] \quad k = 8 \\
[-17, 26, 0]
\end{array}
\]

Thus $\text{gcd}(51, 78) = 3$.

• (5 points) $63^{-1} \mod 97$. The extended Euclidean algorithm computes:

\[
\begin{array}{c}
[1, 0, 97] \\
[0, 1, 63] \quad k = 1 \\
[1, -1, 34] \quad k = 1 \\
[1, -1, 34] \quad k = 1 \\
[2, -3, 5] \quad k = 5 \\
[-11, 17, 4] \quad k = 1 \\
[13, -20, 1] \quad k = 4 \\
[-63, 97, 0]
\end{array}
\]

Thus $63^{-1} = -20 = 77 \mod 97$.

• (5 points) $25^{-1} \mod 45$. Since $\text{gcd}(25, 45) = 5$, $25^{-1} \mod 45$ doesn’t exist.

• (5 points) $7^{134} \mod 10$. The binary-exponentiation algorithm computes

\[
\begin{align*}
7^1 &= 7 \mod 10 \\
7^2 &= 9 \mod 10 \\
7^4 &= 1 \mod 10 \\
7^8 &= 1 \mod 10 \\
7^{16} &= 1 \mod 10 \\
7^{32} &= 1 \mod 10 \\
7^{64} &= 1 \mod 10 \\
7^{128} &= 1 \mod 10
\end{align*}
\]

so $7^{134} = 7^{128} \times 7^4 \times 7^2 = 1 \times 1 \times 9 = 9 \mod 10$. 


Problem 2

**RSA** Suppose Alice generates the RSA key-pair $P_A = (28, 65)$, $S_A = (7, 65)$ and Bob generates key-pair $P_B = (15, 77)$, $S_B = (36, 77)$. Alice sends the message 8 to Bob by encrypting and then signing.

**Solutions in-line**

- (5 points) What key does she use to encrypt? $P_B$.
- (5 points) What key does she use to sign? $S_A$.
- (5 points) What is the result she sends to Bob? Assume she doesn’t use any padding or hashing when encrypting or signing.

To encrypt, Alice needs to compute $8^{15} = 43 \mod 77$. She can compute this using binary exponentation. To sign, she must compute $43^7 = 17 \mod 65$. 
Problem 3

Hash Functions (20 points) Recall that a hash function $H$ maps arbitrary-length inputs to a fixed-size output. A hash function is insecure if, given a value $y$, it is easy to compute an input $x$ such that $H(x) = y$. Show that the hash function

\[
H(m) \\
\text{let } (m_0, \ldots, m_t) = m; \quad \text{// divide } m \text{ into 128-bit blocks, padded if necessary} \\
\text{let } c = m_0; \\
\text{for } i = 1 \text{ to } t \\
\quad c = \text{AES}(c, m_i); \\
\text{return } c;
\]

is insecure by describing a procedure for quickly computing, for any value $y$, an input $x$ such that $H(x) = y$. Hint: What is $H((m_0, m_1))$? Given $y$, can you find a two-block message $m = (m_0, m_1)$ such that $H(m) = y$?

Solution $H(m_0) = \text{AES}(0, m_0)$, so given $t$, the message $m = \text{AES}^{-1}(0, t)$ has $H(m) = t$. 
Problem 4

CBC Mode (20 points) Suppose an attacker intercepts a message $C = (IV, C_1, C_2, \ldots, C_n)$ encrypted with CBC-mode. The attacker wants to modify $C$ to get $C'$ and send $C'$ on to the intended recipient for $C$. The receiver will then compute $M' = D(k, C')$. Show how the attacker can change any bit he chooses in the first block of $M'$ by modifying the $IV$ in $C$.

Solution In CBC mode, $m_1 = E^{-1}(k, c_1) \oplus IV$, so flipping a bit of the IV will flip the corresponding bit in $m_0$. 
Problem 5

**Diffie-Helman** (20 points) We saw in class that Diffie-Hellman key agreement is vulnerable to a man-in-the-middle attack. Draw a modified version of the Diffie-Hellman protocol that uses public-key signatures to prevent the man-in-the-middle attack.

**Solution** The original Diffie-Hellman protocol proceeds as follows:

\[
\begin{align*}
A : & \text{ pick random } a \\
A \rightarrow B : & \quad x = g^a \\
B : & \text{ pick random } b \\
B \rightarrow A : & \quad y = g^b \\
A : & \quad k = y^a \\
B : & \quad k = x^b
\end{align*}
\]

To prevent an attacker from impersonating A or B, they can each sign their messages:

\[
\begin{align*}
A : & \text{ pick random } a \\
A \rightarrow B : & \quad x = g^a, s = \text{Sig}(S_A, x) \\
B : & \text{ if Vrfy}(P_A, x, s) \text{ fails, then abort} \\
B : & \text{ pick random } b \\
B \rightarrow A : & \quad y = g^b, t = \text{Sig}(S_B, y) \\
A : & \text{ if Vrfy}(P_B, y, y) \text{ fails, then abort} \\
A : & \quad k = y^a \\
B : & \quad k = x^b
\end{align*}
\]