Problem 1. (10 points) Give a clear, concise, and complete description of the **RSA Public Key Cryptosystem**, including a brief indication of why it works.

Solution.

**Key Creation**: Alice’s private key is a pair of large primes \( p \) and \( q \). Her public key is the product \( N = pq \), together with an encryption exponent \( e \).

**Encryption**: Bob chooses a plaintext \( m \) modulo \( N \). His ciphertext is the quantity \( c = m^e \pmod{N} \).

**Decryption**: Alice first computes an inverse \( d \equiv e^{-1} \pmod{\phi(N)} \), where \( \phi(N) = (p-1)(q-1) \). She then decrypts Bob’s ciphertext by computing \( c^d \pmod{N} \). For added efficiency, she can instead take \( d \) to be the smaller number given by

\[
d = e^{-1} \pmod{\gcd(p-1, q-1)}.
\]

**Why It Works**: Euler’s theorem tells us that \( a^{\phi(N)} \equiv 1 \pmod{N} \) (provided that \( \gcd(a, N) = 1 \)). Since \( de = 1 + n(p-1)(q-1) \) for some \( n \), Alice computes

\[
c^d \equiv m^d \equiv m \pmod{N}.
\]

Problem 2. (10 points) Give a clear, concise, and complete description of the **RSA Digital Signature Scheme**, including a brief indication of why it works.

Solution.

**Key Creation**: Alice’s private key is a pair of large primes \( p \) and \( q \). Her public key is the product \( N = pq \), together with a verification exponent \( e \).

**Signing**: Alice first computes an inverse \( d \equiv e^{-1} \pmod{\phi(N)} \). Then her signature on a document \( D \pmod{N} \) is the number \( S \equiv D^d \pmod{N} \).

**Verification**: Bob computes \( S^e \pmod{N} \) and verifies that it is equal to \( D \).

**Why It Works**: As in Problem 1, we have \( de = 1 + n(p-1)(q-1) \) for some \( n \), so Bob computes

\[
S^e \equiv D^{de} = D^{1+n(p-1)(q-1)} = D \cdot (D^{(p-1)(q-1)})^n \equiv D \cdot 1^n \equiv D \pmod{N}.
\]
**Problem 3.** (10 points) Describe briefly and clearly an algorithm that has a high probability of solving the discrete logarithm problem
\[ g^x \equiv a \pmod{p} \]
in a subexponential number of steps. (You may assume that \( g \) has known order \( n \), where \( n \) is prime.)

**Solution.** The Index Calculus is a subexponential algorithm to solve the DLP. Compute random powers \( g^k \mod{p} \) and find a lot of them that are \( B \)-smooth. In other words, find values of \( k \) so that \( g^k \mod{p} \) factors as
\[ g^k \equiv \prod_{\ell \text{ prime} \atop \ell \leq B} \ell^{e_{k,\ell}} \pmod{p}. \]
Taking the discrete logarithm (to the base \( g \)) of both sides gives the relation
\[ k \equiv \sum_{\ell \text{ prime} \atop \ell \leq B} e_{k,\ell} \log_g(\ell) \pmod{n}. \]
Note that we know the values of the \( e_{k,\ell} \). We keep doing this until we find \( \pi(B) \) of these relations, and then we can use linear algebra to solve for the unknown quantities \( \log_g(\ell) \) for all primes \( \ell \leq B \). Next we try random values of \( j \) until we find some \( a g^{-j} \mod{p} \) that is \( B \)-smooth, say
\[ a g^{-j} \equiv \prod_{\ell \text{ prime} \atop \ell \leq B} \ell^{f_{\ell}}. \]
Then
\[ \log_g(a) = j + \sum_{\ell \text{ prime} \atop \ell \leq B} f_{\ell} \log_g(\ell). \]
Taking \( B \) to be a small power of \( L(B) = e^{\sqrt{\ln B \ln \ln B}} \) gives an expected running time that is also a small power of \( L(B) \), so the algorithm is subexponential.

**NOTE:** The Babystep-Giantstep algorithm and Pollard’s rho algorithm can be used to solve the DLP, but their expected running time is exponential, not subexponential.

**Problem 4.** (10 points) As in the previous problem, assume that \( g \) has known order \( n \) modulo \( p \), where \( n \) is prime. Describe briefly and clearly an algorithm that has a high probability of solving the discrete logarithm problem
\[ g^x \equiv a \pmod{p} \]
in \( O(\sqrt{n}) \) steps while using very little storage.
Solution. Pollard’s rho algorithm works as follows. Define a function

\[ f(x) = \begin{cases} 
  gx & \text{if } 1 \leq x < \frac{p}{3}, \\
  x^2 & \text{if } \frac{p}{3} \leq x < \frac{2p}{3}, \\
  ax & \text{if } \frac{2p}{3} \leq x < p.
\end{cases} \]

Starting with \( x_0 = y_0 = 1 \), compute

\[ x_{i+1} = f(x_i) \quad \text{and} \quad y_{i+1} = f(f(y_i)) \quad \text{for } i = 0, 1, 2, \ldots. \]

Continue until finding a match \( x_k = y_k \). The expected time to the first match (as we proved in class) is \( k \approx 1.25 \sqrt{n} \).

It is clear from the definition of \( f \) that \( x_i \) and \( y_i \) can be written in the form

\[ x_i = a^{\alpha_i} g^{\beta_i} \quad \text{and} \quad y_i = a^{\gamma_i} g^{\delta_i}. \]

While computing the sequences of \( x_i \)'s and \( y_i \)'s, we also compute sequences \( \alpha_i, \beta_i, \gamma_i, \delta_i \) as follows. We start with \( \alpha_0 = \beta_0 = \gamma_0 = \delta_0 = 0 \), and then we compute successive values using the formulas

\[ \alpha_{i+1} = \begin{cases} 
  \alpha_i & \text{if } 1 \leq x_i < \frac{p}{3}, \\
  2\alpha_i & \text{if } \frac{p}{3} \leq x_i < \frac{2p}{3}, \\
  \alpha_i + 1 & \text{if } \frac{2p}{3} \leq x_i < p,
\end{cases} \]

\[ \beta_{i+1} = \begin{cases} 
  \beta_i + 1 & \text{if } 1 \leq x_i < \frac{p}{3}, \\
  2\beta_i & \text{if } \frac{p}{3} \leq x_i < \frac{2p}{3}, \\
  \beta_i & \text{if } \frac{2p}{3} \leq x_i < p.
\end{cases} \]

The computation of \( \gamma_{i+1} \) and \( \delta_{i+1} \) is similar, essentially doing the formulas for \( \alpha \) and \( \beta \) twice, once with \( y_i \) and once with \( f(y_i) \).

Once we find a collision \( x_k = y_k \), then we have

\[ a^{\alpha_k} g^{\beta_k} \equiv a^{\gamma_k} g^{\delta_k} \pmod{p}. \]

Hence

\[ g^{\beta_k - \delta_k} \equiv a^\gamma g^{\delta_k} \pmod{p}, \]

which implies (taking discrete logarithms to the base \( g \))

\[ \beta_k - \delta_k \equiv (\gamma - \alpha_k) \log_g(a) \pmod{p-1}. \]

If \( \gcd(\gamma_i - \alpha_k, p - 1) = 1 \), we can immediately solve for \( \log_g(a) \).

In general, let \( G = \gcd(\gamma_i - \alpha_k, p - 1) \). Then we can solve

\[ \frac{\beta_k - \delta_k}{G} \equiv \frac{\gamma_i - \alpha_k}{G} \log_g(a) \pmod{p-1}. \]
for
\[ A = \log_g(a) \pmod{G}. \]
Then, since generally \( G \) won’t be too large, we can try each of
\[ A, \quad A + \frac{p-1}{G}, \quad A + 2\frac{p-1}{G}, \ldots \quad A + (G-1)\frac{p-1}{G} \]
in the equation \( g^x \equiv a \pmod{p} \) to find the solution.

**Problem 5.** (9 points) Let \( k > 0 \) be a fixed constant. Consider the function
\[ F(x) = 2^{x^k} \]
(a) For what values of \( k \) does the function \( F(x) \) have subexponential growth (as a function of \( x \))?
(b) For what values of \( k \) does the function \( F(x) \) grow at least exponentially (as a function of \( x \))?  
(c) For what values of \( k \) does the function \( F(x) \) have polynomial growth (as a function of \( x \))?

**Solution.** (a) The function \( F(x) \) is subexponential if it grows slower than \( C^x \) for every \( C > 1 \). So we need \( 2^{x^k} < C^x \) when \( x \) is sufficiently large. Taking logarithms, we need \( x^k \log(2) < x \log(C) \), and this needs to be true for every \( C > 1 \). So \( F(x) \) is subexponential if and only if \( k < 1 \).

(b) \( F(x) \) grows exponentially if there is some \( C > 1 \) such that \( F(x) > C^x \) when \( x \) is sufficiently large, so we need \( 2^{x^k} > C^x \). Taking logarithms, we need there to be some \( C > 1 \) so that \( x^k \log(2) > x \log(C) \). Taking (say) \( C = \frac{3}{2} \), this is true provided \( k \geq 1 \), while it is never true if \( k < 1 \). So \( F(x) \) has exponential growth if and only if \( k \geq 1 \).

(c) \( F(x) \) has polynomial growth if there is some \( d > 0 \) such that \( F(x) < x^d \) when \( x \) is sufficiently large. Taking logarithms, we need there to be some \( d \) such that \( x^k \log(2) < d \log(x) \). This is never true for large values of \( x \), since if \( k > 0 \), then \( x^k \) grows faster than \( \log(x) \). So \( F(x) \) has polynomial growth for no values of \( k > 0 \).

**Problem 6.** (10 points) Let \( E \) be the elliptic curve
\[ E : y^2 = x^3 + x + 6. \]
The points \( P = (2, 4) \) and \( Q = (3, 6) \) are in \( E(\mathbb{Q}) \).
(a) Compute the sum \( P \oplus Q \) using the group law on \( E \).
(b) Compute the sum \( Q \oplus Q \) using the group law on \( E \).
Solution. If you’ve memorized the elliptic curve addition formulas, you can use those. But it’s not necessary, you can use the definition in terms of intersections.

(a) The slope of the line \( L \) through \( P \) and \( Q \) is \( \frac{6-4}{3-2} = 2 \), so \( L \) is the line \( y - 4 = 2(x - 2) \), so

\[
L : y = 2x.
\]

To find \( E \cap L \), we substitute the equation of \( L \) into the equation of \( E \), which gives

\[
4x^2 = x^3 + x + 6.
\]

Since we know that \( x = 2 \) and \( x = 3 \) are roots, we have

\[
x^3 - 4x^2 + x + 6 = (x - 2)(x - 3)(x - x_3).
\]

Comparing the constant terms, for example, gives \( x_3 = -1 \), and then substituting into the equation for \( L \) gives \( y_3 = -2 \). So the third point on \( E \cap L \) is \((-1, -2)\). Then to compute \( P \oplus Q \), we change the sign on the \( y \)-coordinate, which gives the answer \( P \oplus Q = (-1, 2) \).

(b) This is similar, but we need the tangent line \( L \) to \( E \) at \( Q \). We have

\[
2y \frac{dy}{dx} = 3x^2 + 1,
\]

so substituting in \( Q \) gives

\[
\left. \frac{dy}{dx} \right|_Q = \frac{3 \cdot 3^2 + 1}{2 \cdot 6} = \frac{7}{3}.
\]

This is the slope of \( L \), so \( L \) is the line

\[
L : y - 6 = \frac{7}{3}(x - 3).
\]

A little algebra gives

\[
L : y = \frac{7}{3}x - 1.
\]

Substituting into the equation for \( E \), we find

\[
\left( \frac{7}{3}x - 1 \right)^2 = x^3 + x + 6.
\]

We know that \( x = 3 \) is a double root, so

\[
x^3 + x + 6 - \left( \frac{7}{3}x - 1 \right)^2 = (x - 3)^2(x - x_3).
\]

It’s not necessary to multiply it all out, we just need to compare coefficients of one power of \( x \). So for example, we can look at the constant terms (which is the same as substituting \( x = 0 \)), which gives

\[
6 - 1^2 = -9x_3.
\]
Hence \( x_3 = -5/9 \). Substituting into the equation for \( L \) gives \( y_3 = \frac{7}{9} \cdot x_3 - 1 = -62/27 \), so \((-5/9, -62/27) \in E \cap L\). Switching the sign of the \( y \)-coordinate gives \( Q \oplus Q = (-\frac{5}{9}, \frac{62}{27}) \).

**Problem 7.** (10 points) There are 10 people in a room. What is the probability that at least two of them have the same birthday? (You may assume that every year has 365 days. Also, you may leave your answer in the form of a product.)

**Solution.** More generally, suppose that there are \( k \) people and that there are \( N \) days in a year. Label the people \( P_1, P_2, \ldots, P_k \) Then

\[
\Pr(\text{at least two of } P_1, \ldots, P_k \text{ are the same}) = 1 - \Pr(\text{all of } P_1, P_2, \ldots, P_k \text{ are different})
\]

\[
= 1 - \Pr(P_2 \neq P_1) \Pr(P_3 \neq P_1, P_2 \mid P_1, P_2 \text{ are distinct}) \ldots
\]

\[
= 1 - \prod_{i=2}^{k} \Pr(P_i \neq P_1, P_2, \ldots, P_{i-1} \mid P_1, P_2, \ldots, P_{i-1} \text{ are distinct})
\]

\[
= 1 - \prod_{i=2}^{k} \frac{N - (i - 1)}{N}
\]

\[
= 1 - \prod_{j=1}^{k-1} \left( 1 - \frac{j}{N} \right).
\]

So for 10 people and 365 days in a year, the probability that at least two of the people share a birthday is

\[
1 - \left( 1 - \frac{1}{365} \right) \left( 1 - \frac{2}{365} \right) \cdots \left( 1 - \frac{9}{365} \right).
\]

(It turns out that this value is approximately 11.695%.)

**Problem 8.** (15 points) Let \( N = pq \) be a product of two large primes. The value of \( N \) is public, but the values of \( p \) and \( q \) are not. Consider the following digital signature scheme.

**Key Creation:** Alice chooses numbers \( k \) and \( a \) and computes

\[ b \equiv k^a \pmod N. \]

Her private signing key is \( k \), and her public verification key is the pair \((a, b)\).

**Signing:** To sign a document \( D \), Alice chooses a random number \( r \) and computes

\[ S \equiv Dk^r \pmod N. \]
The document and signature consists of the triple \((D, r, S)\).

**Verification**: Bob accepts that the signature is valid if

\[ S^a \equiv D^a b^r \pmod{N}. \]

(a) Prove that if Alice has signed \(D\), then verification works.

(b) Suppose that Eve is given a signed document \((D, r, S)\). Explain how Eve can create a signature on another document \(D'\) without knowing Alice’s private signing key.

(c) Suppose that Eve is given two signed documents \((D_1, r_1, S_1)\) and \((D_2, r_2, S_2)\), and suppose further that Alice’s chosen random numbers \(r_1\) and \(r_2\) happen to satisfy \(\gcd(r_1, r_2) = 1\).

Explain how Eve can compute Alice’s private signing key \(k\).

**Solution.**

(a) We have

\[ S^a \equiv (Dk^a)^a \equiv D^a k^{ra} \equiv D^a b^r \pmod{N}. \]

(b) The triple \((D', r, D'D^{-1} S)\) is a valid signature on \(D'\), where \(D^{-1}\) means the inverse of \(D\) modulo \(N\). To check that \((D', r, D'D^{-1} S)\) is a valid signature on \(D'\), we compute

\[ (D'D^{-1} S)^a = (D')^a \cdot D^{-a} \cdot S^a \]
\[ \equiv (D')^a \cdot D^{-a} \cdot (D^a b^r) \pmod{N} \]
\[ \equiv (D')^a b^r \pmod{N}. \]

(c) Since \(\gcd(r_1, r_2) = 1\), Eve can find integers \(u_1, u_2\) such that \(r_1 u_1 + r_2 u_2 = 1\). Then

\[ S_1^{u_1} \cdot S_2^{u_2} \equiv (D_1 k^{r_1})^{u_1} \cdot (D_2 k^{r_2})^{u_2} \pmod{N} \]
\[ \equiv D_1^{u_1} \cdot D_2^{u_2} k^{r_1 u_1 + r_2 u_2} \pmod{N} \]
\[ \equiv D_1^{u_1} \cdot D_2^{u_2} k \pmod{N}. \]

Since Eve knows all of the quantities \(S_1, S_2, D_1, D_2, u_1, u_2\), she can compute

\[ k \equiv S_1^{u_1} \cdot S_2^{u_2} \cdot D_1^{-u_1} \cdot D_2^{-u_2} \pmod{N}. \]

**Problem 9.** (10 points) For each part, circle the answer and fill in the box to correctly complete the statements.

(a) To **factor an integer**, the running time of the best known algorithm is **Subexponential**. This algorithm is called the **Number Field** algorithm.
Sieve. (Other subexponential factorization algorithms are the Quadratic Sieve and Lenstra’s Elliptic Curve Factorization Algorithm. Any of these are acceptable answers.)

(b) To solve the discrete logarithm problem in \((\mathbb{Z}/p\mathbb{Z})^*\), the running time of the best known algorithm is Subexponential. This algorithm is called the Index Calculus.

(c) To compute \(\gcd(a, b)\), the running time of the best known algorithm is Polynomial. This algorithm is called the Extended Euclidean Algorithm.

(d) To solve the elliptic curve discrete logarithm problem, the running time of the best known algorithm is Exponential. This algorithm is called the Pollard’s rho method. Other acceptable answers are Babystep-Giantstep method, or simply Collision algorithms.

(e) To compute \(a^n \pmod{N}\), the running time of the best known algorithm is Polynomial. This algorithm is called Square-and-Multiply.

Problem 10. (9 points) A private key cryptosystem has three keys, three plaintexts, and four ciphertexts:

\[ \mathcal{K} = \{k_1, k_2, k_3\}, \quad \mathcal{M} = \{m_1, m_2, m_3\}, \quad \mathcal{C} = \{c_1, c_2, c_3, c_4\}. \]

(a) Is the following a valid encryption table for \((\mathcal{K}, \mathcal{M}, \mathcal{C})\)? If not, then why not?

<table>
<thead>
<tr>
<th></th>
<th>(m_1)</th>
<th>(m_2)</th>
<th>(m_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(k_1)</td>
<td>(c_1)</td>
<td>(c_3)</td>
<td>(c_4)</td>
</tr>
<tr>
<td>(k_2)</td>
<td>(c_2)</td>
<td>(c_4)</td>
<td>(c_2)</td>
</tr>
</tbody>
</table>

(b) Is the following a valid encryption table for \((\mathcal{K}, \mathcal{M}, \mathcal{C})\)? If not, then why not?

<table>
<thead>
<tr>
<th></th>
<th>(m_1)</th>
<th>(m_2)</th>
<th>(m_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(k_1)</td>
<td>(c_1)</td>
<td>(c_2)</td>
<td>(c_4)</td>
</tr>
<tr>
<td>(k_2)</td>
<td>(c_2)</td>
<td>(c_4)</td>
<td>(c_1)</td>
</tr>
</tbody>
</table>

(c) Is the following a valid encryption table for \((\mathcal{K}, \mathcal{M}, \mathcal{C})\)? If not, then why not?

<table>
<thead>
<tr>
<th></th>
<th>(m_1)</th>
<th>(m_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(k_1)</td>
<td>(c_1)</td>
<td>(c_3)</td>
</tr>
<tr>
<td>(k_2)</td>
<td>(c_2)</td>
<td>(c_4)</td>
</tr>
</tbody>
</table>

Solution. (a) \[\text{No}\]. Because the encryption function \(e_{k_2}\) is not one-to-one, since \(e_{k_2}(m_1) = e_{k_2}(m_3) = c_2\).

(b) Yes, this table gives a well-defined encryption function for each key.
(c) No. Because the plaintext $m_3$ isn’t assigned a ciphertext, so the domain of $e_{k_1}$ and $e_{k_2}$ is not all of $\mathcal{M}$.

**Problem 11.** (10 points) A private key cryptosystem has three keys, three plaintexts, and four ciphertexts:

$$\mathcal{K} = \{k_1, k_2, k_3\}, \quad \mathcal{M} = \{m_1, m_2, m_3\}, \quad \mathcal{C} = \{c_1, c_2, c_3, c_4\}.$$ 

Its encryption function is given by the table

<table>
<thead>
<tr>
<th></th>
<th>$m_1$</th>
<th>$m_2$</th>
<th>$m_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_1$</td>
<td>$c_1$</td>
<td>$c_3$</td>
<td>$c_4$</td>
</tr>
<tr>
<td>$k_2$</td>
<td>$c_4$</td>
<td>$c_3$</td>
<td>$c_2$</td>
</tr>
</tbody>
</table>

The probability that Alice chooses a certain key or a certain plaintext is given by

$$f(k_1) = \frac{1}{3}, \quad f(k_2) = \frac{2}{3}, \quad f(m_1) = \frac{1}{2}, \quad f(m_2) = \frac{3}{8}, \quad f(m_3) = \frac{1}{8}.$$ 

And as usual, her choice of key and her choice of plaintext are independent.

(a) What are the values of the following four probabilities? Just give the value, you do not need to explain why it is correct.

$$f_{\mathcal{C}|\mathcal{M}}(c_1 \mid m_1) = \boxed{\text{---}} \quad f_{\mathcal{M}|\mathcal{C}}(m_1 \mid c_1) = \boxed{\text{---}}$$

$$f_{\mathcal{M}|\mathcal{C}}(m_2 \mid c_3) = \boxed{\text{---}} \quad f_{\mathcal{K}|\mathcal{C}}(k_2 \mid c_1) = \boxed{\text{---}}$$

(b) Compute the value of $f_{\mathcal{C}}(c_4)$.

(c) Compute the key entropy $H(K)$ and the plaintext entropy $H(M)$. (You may leave your answers as sums.)

(*) Bonus (worth 1 point): Compute the value of $f_{\mathcal{K}|\mathcal{C}}(k_2 \mid c_4)$.

**Solution.** (a)

$$f_{\mathcal{C}|\mathcal{M}}(c_1 \mid m_1) = f_{\mathcal{K}}(k_1) = \frac{1}{3}$$

$$f_{\mathcal{M}|\mathcal{C}}(m_1 \mid c_1) = 1$$

$$f_{\mathcal{M}|\mathcal{C}}(m_2 \mid c_3) = 1$$

$$f_{\mathcal{K}|\mathcal{C}}(k_2 \mid c_1) = 0$$
(b) 
\[ f_C(c_4) = f_K(k_1) f_M(d_{k_1}(c_4)) + f_K(k_2) f_M(d_{k_2}(c_4)) \]
\[ = f_K(k_1) f_M(m_3) + f_K(k_2) f_M(m_1) \]
\[ = \frac{1}{3} \cdot \frac{1}{5} + \frac{2}{3} \cdot \frac{1}{2} \]
\[ = \frac{3}{8} \]

(c) The entropy \( H(X) \) of a random variable whose probabilities are \( f_X(\omega_1) = p_1, \ldots, f_X(\omega_n) = p_n \) is by definition the quantity
\[ H(X) = -p_1 \log_2(p_1) - p_2 \log_2(p_2) - \cdots - p_n \log_2(p_n). \]

So
\[ H(K) = -\frac{1}{3} \log_2 \left( \frac{1}{3} \right) - \frac{2}{3} \log_2 \left( \frac{2}{3} \right). \]

There are various ways to simplify this, for example, it is equal to \( \frac{1}{3} \log_2 \left( \frac{25}{4} \right) \). It is approximately equal to 0.918.

Similarly,
\[ H(M) = -\frac{1}{2} \log_2 \left( \frac{1}{2} \right) - \frac{3}{8} \log_2 \left( \frac{3}{8} \right) - \frac{1}{8} \log_2 \left( \frac{1}{8} \right). \]

Again, there are various ways to simplify the answer, for example \( H(M) = 2 - \frac{3}{8} \log_2(3) \). It is approximately equal to 1.406.

(*) Bonus Problem:
\[ f_{K|C}(k_2 \mid c_4) = \frac{f_{K,C}(k_2, c_4)}{f_C(c_4)} \]
\[ = \frac{f_{K,M}(k_2, m_1)}{f_C(c_4)} \] since the choice of key and plaintext are independent,
\[ = \frac{\frac{2}{3} \cdot \frac{1}{2}}{\frac{3}{8}} \] using part (a),
\[ = \frac{2}{5} \]