Problem 1. Cramer-Shoup cryptosystem

1. Cramer–Shoup cryptosystem is an asymmetric key encryption algorithm, which is proven to be secure against adaptive chosen ciphertext attack. It consists of three algorithms: the key generator, the encryption algorithm, and the decryption algorithm.

   Key generator

   - Alice generates a cyclic group G of order q and finds two generators g_1, g_2.
   - Alice randomly chooses x_1, x_2, y_1, y_2, z from $\{0, \ldots, q-1\}$.
   - Alice computes $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$ and $h = g_1^z$.
   - Alice publishes $(c, d, h, G, q, g_1, g_2)$ as the public key and keeps (x_1, x_2, y_1, y_2, z) as the private key.

   Encryption

   - Bob converts m into an element of G and choose a random k from $\{0, \ldots, q-1\}$.
   - Bob computes $u_1 = g_1^k$, $u_2 = g_2^k$, $e = h^k m$, $\alpha = H(u_1, u_2, e)$ where $H(x)$ is a collision-resistant cryptographic hash function, and $v = c^k d^{k\alpha}$
   - Bob sends the ciphertext (u_1, u_2, e, v) to Alice.

   Decryption

   - Alice computes $\alpha = H(u_1, u_2, e)$ and verifies that $u_1^{x_1} u_2^{x_2} (u_1^{y_1} u_2^{y_2})^{\alpha} = v$, else the decryption algorithm ends with failure output.
   - Otherwise she computes the plaintext $m = e/h^k$.
   - The decryption stage correctly decrypts any properly-formed ciphertext, since $u_1^z = g_1^{kz} = h^k$.

2. Adaptive chosen ciphertext attacks can be applied if a ciphertext can be modified in specific ways that will have a predictable effect on the decryption of that message. However, The decryption algorithm of Cramer-Shoup cryptosystem rejects all invalid ciphertexts constructed by an attacker through verifying the result generated by a collision-resistant cryptographic hash function. It limits ciphertext malleability so that it can be considered secure under this kind of attack.

3. (a) Similarities: Both are public key cryptosystems computed in a cyclic group G, the private keys are both based on the difficulty of solving Discrete Logarithm Problem.

   (b) Differences: Cramer–Shoup cryptosystem consists a collision-resistant cryptographic hash function which is used to verify the ciphertext while Elgamal cryptosystem doesn't.

Problem 2. Simple Questions

1. $h(x)$ is not a good hash function because it is not second pre-image resistant and collision resistant. Given p a prime and $p \nmid a$, we have $gcd(a,p) = 1, \alpha^{p-1} \equiv 1 \bmod p$. Therefore, knowing x, we can forge $x' = x + k(p-1), k \in \mathbb{Z}$ that $h(x) = h(x')$. Or we can directly give $x_1, x_2 = x_1 + k(p-1), k \in \mathbb{Z}$ so that $h(x_1) = h(x_2)$.

2. Here gives the result

$$\lfloor 2^{30}\sqrt{2} \rfloor_{16} = 5A827999$$
$$\lfloor 2^{30}\sqrt{3} \rfloor_{16} = 6ED9EBA1$$
$$\lfloor 2^{30}\sqrt{5} \rfloor_{16} = 8F1BBCDC$$
$$\lfloor 2^{30}\sqrt{10} \rfloor_{16} = CA62C1D6$$

The constants $K_0, ..., K_{79}$ are defined by

$$K_i = \begin{cases} 5A827999 & \text{if } 0 \leq i \leq 19 \\ 6ED9EBA1 & \text{if } 20 \leq i \leq 39 \\ 8F1BBCDC & \text{if } 40 \leq i \leq 59 \\ CA62C1D6 & \text{if } 60 \leq i \leq 79 \end{cases}$$

Compare the results above, we found that they are identical each corresponding line.

Problem 3. Birthaday Paradox

1. Given $g(x) = \ln(1-x) + x + x^2$,

$$g'(x) = -\frac{1}{1-x} + 1 + 2x = 0$$
$$x_1 = 0, x_2 = \frac{1}{2}$$
$$g''(x) = -\frac{1}{(x-1)^2} + 2$$
$$g''(0) = 1, \text{ local minimum}$$
$$g''\left(\frac{1}{2}\right) = -2, \text{ local maximum}$$

Since $g(0) = 0$, we can conclude that when $x \in \left[0, \frac{1}{2}\right]$, $g(x) \geqslant 0$.

Let $h(x) = \ln(1-x) + x$

$$h'(x) = -\frac{1}{1-x} + 1 = 0$$
$$x = 0$$
$$h''(x) = -\frac{1}{(x-1)^2}$$

$$h''(0) = -1, \text{ local maximum}$$

Since $h(0) = 0$, we can conclude that when $x \in \left[0, \frac{1}{2}\right]$, $h(x) \leqslant 0$

In conclusion, we have proved $-x - x^2 \leqslant \ln(1-x) \leqslant -x$

2. Given $r \leqslant \frac{n}{2}$ and $j \in [1, r-1]$, thus $\frac{j}{n} \in \left[0, \frac{1}{2}\right]$. Based on (1), we have

$$-\frac{j}{n} - \left(\frac{j}{n}\right)^2 \leqslant \ln\left(1 - \frac{j}{n}\right) \leqslant -\frac{j}{n}$$

$$\sum_{j=1}^{r-1}\left[-\frac{j}{n} - \left(\frac{j}{n}\right)^2\right] \leqslant \sum_{j=1}^{r-1}\ln\left(1 - \frac{j}{n}\right) \leqslant \sum_{j=1}^{r-1} -\frac{j}{n}$$

$$-\frac{(r-1)r}{2n} - \frac{(r-1)r(2r-1)}{6n^2} \leqslant \sum_{j=1}^{r-1}\ln\left(1 - \frac{j}{n}\right) \leqslant -\frac{(r-1)r}{2n}$$

$$\frac{(r-1)r(2r-1)}{6n^2} = \frac{r^3 - \frac{3}{2}r^2 + r}{3n^2} < \frac{r^3}{3n^2}$$

$$-\frac{(r-1)r}{2n} - \frac{r^3}{3n^2} \leqslant \sum_{j=1}^{r-1}\ln\left(1 - \frac{j}{n}\right) \leqslant -\frac{(r-1)r}{2n}$$

3. Exponentiate the inequation above, we have

$$\exp\left(-\frac{(r-1)r}{2n} - \frac{r^3}{3n^2}\right) \leqslant \prod_{j=1}^{r-1}\left(1 - \frac{j}{n}\right) \leqslant \exp\left(-\frac{(r-1)r}{2n}\right)$$

Given $\lambda = \frac{r^2}{2n}$, $c_1 = \sqrt{\frac{\lambda}{2}} - \frac{(2\lambda)^{3/2}}{3}$ and $c_2 = \sqrt{\frac{\lambda}{2}}$

$$-\lambda + \frac{c_1}{\sqrt{n}} = -\frac{r^2}{2n} + \frac{r}{2n} - \frac{r^3}{n^2} = -\frac{(r-1)r}{2n} - \frac{r^3}{3n^2}$$

$$-\lambda + \frac{c_2}{\sqrt{n}} = -\frac{r^2}{2n} + \frac{r}{2n} = -\frac{(r-1)r}{2n}$$

Therefore we have

$$e^{-\lambda}e^{c_1/\sqrt{n}} \leqslant \prod_{j=1}^{r-1}\left(1 - \frac{j}{n}\right) \leqslant e^{-\lambda}e^{c_2/\sqrt{n}}$$

4. When n is large and λ is constnt and is less than $\frac{n}{8}$

$$\lambda = \frac{r^2}{2n} < \frac{n}{8},\ r < \frac{n}{2}$$

$$\lim_{n\to\infty} e^{c_1/\sqrt{n}} = \lim_{n\to\infty} e^{c_2/\sqrt{n}} = \lim_{n\to\infty} e^0 = 1$$

Therefore we can conclude that

$$\prod_{j=1}^{r-1}\left(1-\frac{j}{n}\right) \approx e^{-\lambda}$$

Problem 4. Birthday Attack

1. Using formula derived from birthday paradox, we have

$$1-\prod_{i=1}^{39}\left(1-\frac{i}{1000}\right) = 0.546$$

2.
$$40\left(\frac{1}{1000}\right)\left(\frac{999}{1000}\right)^{39} = 0.0385$$

3. (1) means that hash function is not collision resistant. (2) means that hash function is second pre-image resistant. As for Alice, she can overcome the problem by changing the message a bit so that Eve can't find a collision for the new message.

Problem 5. Faster multiple modular exponentiation

1. The complexity of computing $\alpha^a \mod n$ is $O(\log a)$, the complexity of computing $\beta^b \mod n$ is $O(\log b)$, so the total time complexity is $O(\log ab)$.

2. The revised square and multiply algorithm is given in **Algorithm 1**

3. According to our algorithm, l times of squaring and multiplications are necessary to compute $\alpha^a\beta^b \mod n$.

4. Please see Readme.

Algorithm 1 Faster Multiple Modular Exponentiation Algorithm

Require: Integers α, a, β, b, n, $a = (a_{k_a-1} \dots a_0)_2$, $b = (b_{k_b-1} \dots b_0)_2$.
Ensure: $\alpha^a \beta^b \mod n$

```
k ← max(k_a, k_b) // k_a, k_b are the length of a and b
result ← 1
for i = k − 1 downto 0 do
    result ← result · result mod n
    if a_i = 1 and b_i = 1 then
        result ← result · αβ mod n
    else if a_i = 1 then
        result ← result · α mod n
    else if b_i = 1 then
        result ← result · β mod n
    end if
end for
return result
```