This homework covers Lectures 8 and 9.

1. Find a collision in each of the following functions.

   (a) $f(x, y) = p(p(x) + y)$, where p is the permutation

x	000	001	010	011	100	101	110	111
$p(x)$	101	011	100	111	110	001	000	010

   (b) $f(x, y) = x - 2y^2 + 3xy \bmod 107$. The inputs x and y are in $\mathbb{Z}_{107}^* = \{1, \ldots, 106\}$.

   (c) $f(x, y) = x^3 y^5 \bmod 253$. The inputs x and y are in $\mathbb{Z}_{253}^*$ (they are taken modulo 253 and coprime to 253).

2. The inputs to the Pedersen hash $3^X 6^Y \bmod 23$ are numbers modulo 11, while its output is a quadratic residue modulo 23. The output can be represented as a number modulo 11 by the usual encoding

   $$h(X, Y) = \text{positive square root of } 3^X 6^Y \text{ modulo } 23.$$

   Eve finds the collision $h(h(1, 1), 3) = h(h(1, 3), 1) = Z$ to the Merkle-Damgård transform of h.

   (a) Calculate $h(1, 1)$ and $h(1, 3)$.

   (b) Calculate Z in two ways. Verify that the answers are consistent.

   (c) Use parts (a) and (b) to find a collision for h'.

   (d) Use the collision from part (c) to solve $3^W = 6 \bmod 23$ for W.

3. Let $h(x)$ be the first 16 bits of SHA3 on input x.

   (a) Let $x < x^*$ be the smallest two (binary representations of positive) numbers that collide under h. How large do you expect x^* to be?

   (b) Find x and x^* in part (a). Assume numbers occupy an integral number of bytes and are represented in big endian convention. You may use the following python implementation of h:

   ```
   import hashlib
   def h(x):
       byte_length = (x.bit_length() + 7) // 8
       val = hashlib.sha3_256(x.to_bytes(byte_length, byteorder="big")).digest()
       return int.from_bytes(val[:2], byteorder="big", signed=False)
   ```

   (c) A password is a number between 1 and $2^{12} - 1 = 4095$. You get hold of the following hashes of four random passwords under h:

   $$h(x_0) = 47593 \quad h(x_1) = 10918 \quad h(x_2) = 1938 \quad h(x_3) = 53483$$

   How many calls to h do you expect to make before you discover all four passwords?

   $$\text{about } 2^{17/2} \qquad \text{about } 2^{12} \qquad \text{about } 2^{16}$$

   (d) Find the four passwords in part (c) using a computer. How many calls to h did you make?

   (e) **(Optional)** How confident can you be that numbers you found in (d) are the true passwords and not some other numbers with the same hash value?

   (f) The passwords are now salted by a unique two-bit prefix. The hashes you now observe are

   $$h(x_0) = 47593 \qquad h(2^{12} + x_1) = 62468 \qquad h(2^{13} + x_2) = 26298 \qquad h(2^{13} + 2^{12} + x_3) = 59817$$

   How does your answer to parts (c) and (d) change?

4. The following are partial constructions of signature schemes. Explain why they cannot be completed into schemes that are both functional and unforgeable.

   (a) (checksum-then-sign) $sk = (x_0, x_1)$, $pk = (f(x_0), f(x_1))$ for a one-way function f, $Ver_{pk}(m, s)$ accepts when $f(s) = pk_{\text{checksum}}(m)$. checksum$(m)$ is the modulo 2 sum of the message bits.

   (b) (RSA-then-hash) $sk = d$, $pk = e$ is an RSA key pair for modulus n, $Sign_d(m) = h(m^d \bmod n)$, where h is an unstructured hash function.

   (c) (RSA-and-hash) $sk = d$, $pk = e$ is an RSA key pair for modulus n, $Ver_e(m, s)$ accepts when $h(s^e) = h(m)$, where h is an unstructured hash function.