**Encoding and Encryption**

**Information Integrity, Hashing, and Authentication**

These problems will give you experience different types of **hash algorithms** and **authentication**.

You are required to **show your work** (particularly on the SHA-2 and SHA-3 problems)

1. This will be a partial walkthrough of a **single round** of SHA-2 – specifically, parts of the two **mixers**.

Recall that SHA-2 represents a **state** as 8 64-bit binary numbers, called **A, B, C, D, E, F, G,** and **H**. You are to use the **last 6 digits of your Banner ID** to create these. Specifically:

* The last 8 bits of block **A** will be the binary representation of the **last two digits** of your Banner ID. The other bits will all be **0**.
* The last 8 bits of block **B** will be the binary representation of the **middle two digits** of your Banner ID. The other bits will all be **0**.
* The last 8 bits of block **C** will be the binary representation of the **first two digits** of your Banner ID. The other bits will all be **0**.
* The last 8 bits of block **E** will be the binary representation of the **last two digits** of your Banner ID. The other bits will all be **0**.
* The last 8 bits of block **F** will be the binary representation of the **middle two digits** of your Banner ID. The other bits will all be **0**.
* The last 8 bits of block **G** will be the binary representation of the **first two digits** of your Banner ID. The other bits will all be **0**.
* The last 8 bits of block **H** will be the binary representation of the **last two digits** of your Banner ID. The other bits will all be **0**.

For example, if your ID were **00123456**, these blocks would be:  
**A**: 0000000000000000000000000000000000000000000000000000000001010110   
**B**: 0000000000000000000000000000000000000000000000000000000000110100 **C**: 0000000000000000000000000000000000000000000000000000000000010010   
**E**: 0000000000000000000000000000000000000000000000000000000001010110   
**F**: 0000000000000000000000000000000000000000000000000000000000110100 **G**: 0000000000000000000000000000000000000000000000000000000000010010   
**H**: 0000000000000000000000000000000000000000000000000000000001010110

You may also assume that **D**, **Wi** and **Ki** are all **0**’s (that is, they will have no effect on the below questions).

* 1. Compute **Majority(A, B, C)**
  2. Compute **Rot(E)**. Note that this will involve **all 64 bits** in the block.
  3. Compute **Conditional(E, F, G)**

1. This problem is a simple walkthrough of the **HMAC** algorithm.   
   In this case:
   * The **key** will be the **binary equivalent** of the last two digits of your Banner ID.  
     For example, if your Banner ID ends in 93, this would be **10010011**.
   * The **message** will be the 8 bit string **01001000**.
   * The “hash” function will be the **OR** of the message blocks. That is, if either corresponding bit is **1**, then the resulting bit is **1**.
   * For example, suppose the key were **01010101**. HMAC would work like this:

Key

IPAD

**00110110**

**⊕**

**01010101**

“hash”

Message

**OR**

**01001000**

**01100011**

Key

OPAD

**01011100**

**11001101**

**⊕**

“hash”

**OR**

Intermediate HMAC

**01101011**

**10010001**

Final HMAC

**11111011**

Derive the final HMAC using **your Banner ID** as the key as described above.

1. This question will be a simple walkthrough of the signature/verification process. For this question:
   * Simple **RSA** will be used.
   * The **public key** will be ***E*** = **3** and the modulus ***n*** = **33**.
   * The **private key** will be ***D*** = **7**.
   * The **message** ***M*** will be the **second to last digit** of your **Banner ID** **+ 2**.  
     For example, if your Banner ID is Y00123456, your message will be 5 + 2 = **7**.
   1. **Sign** your message with the private key. What is the resulting signature?
   2. Suppose that you send the message and the signature. However, suppose that Darth intercepts that message and **adds 1 to both the message and the signature**. For example, if the message was **5** and the signature was **11**, Darth would change it to **6** and **12**.  
        
      Use the public key to **verify** that the message no longer matches the signature.
2. We have seen a number of algorithms for **challenge-response** (including symmetric key, public key, and digital signature).  
     
   Suppose that Alice and Bob have a **keyed hash** algorithm (that is, a MAC) with a key that they both know. Describe how they could **implement challenge-response** using their keyed hash algorithm.   
     
   Specifically, suppose that Alice sent a request to Bob, and Bob needed to verify her identity. What would be the **steps** in this process? Specifically, what information would be sent back and forth between Alice and Bob?

1. This will be a partial walkthrough of a **single round** of SHA-3 – specifically, the functions **θ, π,** and **χ**.

Recall that SHA-3 represents a state as a three-dimensional 5 x 5 x ***L*** array. To simplify things, we will use ***L*** = **1** for the number of slices – that is, your answer will be a **single 5 x 5 two dimensional** slice.

You will encode the last two digits of your **Banner ID** in that slice for each of the below problems. Specifically:

* The first, second, and third rows will **all be 0**.
* The **fourth row** of the first slice will be the binary representation of the **seventh digit** of your Banner ID.
* The **fifth row** of the first slice will be the binary representation of the **eighth digit** of your Banner ID.

For example, if your ID were **00123456**, the slice **S** would be:  
0 0 0 0 0  
0 0 0 0 0  
0 0 0 0 0  
0 0 1 0 1  
0 0 1 1 0

* 1. Compute **θ (S)**
  2. Compute **π (S)**
  3. Compute **χ (S)**

Note: You will apply *each of these directly to your original* **S**. That is, you will not be running the result of **θ** through **π**, etc.