Question 1

0 / 2 pts

This problem will test your understanding of the sponge construction by having you simulate it. The internal function used will be the permutation p : $\{0,1\}^8 \rightarrow \{0,1\}^8$ where p(x) = x <<< 1 (an 8-bit permutation where x is rotated left one bit). We will use rate R = 3 bits and capacity C = 5 bits. This setup will be the same for the next few questions.

When data is input to the hash function, the first step is to 10*1 pad the data. For each of the following hash inputs, what would the data become after padding? Include both the data and the padding in your answer.

Give each answer as a sequence of bits without spaces or other characters (ie, use the characters 0 and 1 for your answers and nothing else).

empty string: ~~10000001~~ 101

010: ~~01010001~~ 010101

0011: ~~00111001~~ 001111

00101: ~~00101101~~ 001011001

Add padding to a multiple of the rate (R)

a) Empty String + padding // need a multiple R = 3

= Empty String + 101

= 101

b) 010 + padding

= 010 + 101

= 010101

c) 0011 + padding

= 0011 + 11

= 001111

d) 00101 + padding

= 00101 + 1001

= 001011001

Question 2

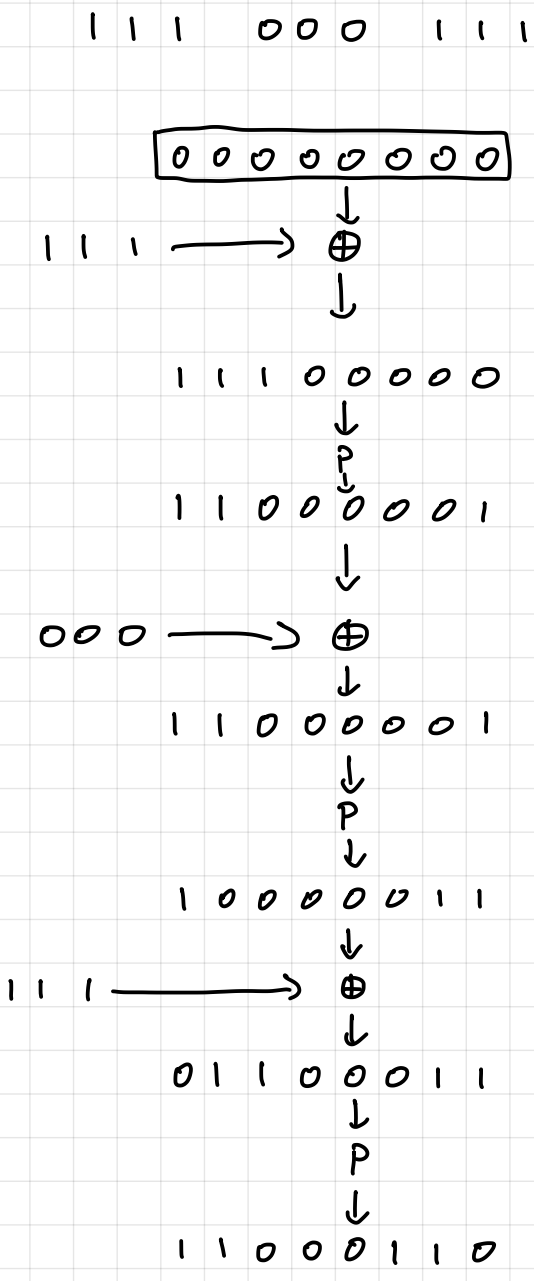
2 / 2 pts

This problem will test your understanding of the sponge construction by having you simulate it. The internal function used will be the permutation $p : \{0,1\}^8 \rightarrow \{0,1\}^8$ where $p(x) = x <<< 1$ (an 8-bit permutation where x is rotated left one bit). We will use rate R = 3 bits and capacity C = 5 bits. This setup is the same as the previous question.

Let's say that after padding your data is 111000111. After absorbing this data, what is the value of your chaining block? (We'll say that absorption ends after the permutation call that includes the padded part of the data and just before any bits are output.)

Give your answer as a sequence of bits without spaces or other characters (ie, use the characters 0 and 1 for your answers and nothing else).

11000110



Question 3

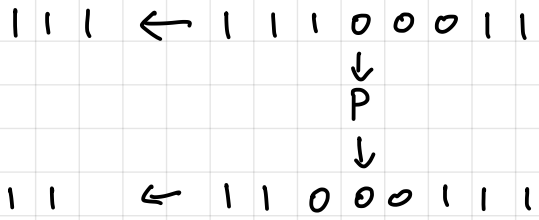
2 / 2 pts

This problem will test your understanding of the sponge construction by having you simulate it. The internal function used will be the permutation p : $\{0,1\}^8 \rightarrow \{0,1\}^8$ where p(x) = x <<< 1 (an 8-bit permutation where x is rotated left one bit). We will use rate R = 3 bits and capacity C = 5 bits. This setup is the same as the previous question.

Let's say that after absorbing all the input data the chaining block is 11100011. If the user of the hash specifies that 5 bits should be output, what are the bits output?

Give your answer as a sequence of bits without spaces or other characters (ie, use the characters 0 and 1 for your answers and nothing else).

11111



Question 4

1 / 2 pts

Choose the dropdown choice that fits best for the statement.

Cryptographic hash functions tend to be much slower than almost-universal hash functions.

much faster?

If you could choose one of the following properties for your cryptographic hash function, which would it be? Second pre-image resistance

Question 5

2 / 2 pts

Recall that divisionless modular reduction computes the mod of 2^a-b without using division. What mod is being performed by the following code snippet? Give your answer by telling me the a and b of the modulus.

```
x = (x >> 17) * 3 + (x & 0x1FFFF);
```

a = 17

b = 3

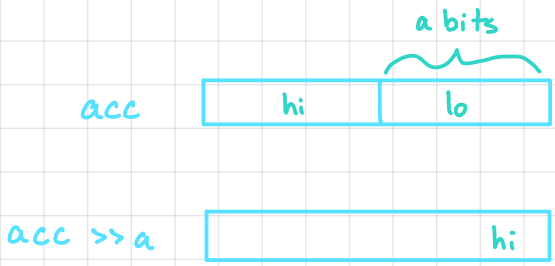
---

divisionless_mod (acc, $p = 2^a - b$)

hi = acc >> a

lo = acc & 0b 1111...111 (a)

return hi * b + lo



Question 6

3 / 3 pts

What number do you get after a single application of divisionless mod on 200 mod 14? Note that the correct answer may not be less than 14.

32

200 mod 14

$2^4 - 2$

$2 \cdot 2 \cdot 2 \cdot 2$

$a = 4$

$b = 2$

200 = 1100 1000

lo

hi

result = hi * b + lo

= 12 * 2 + 8

= 24 + 8

= 32

Question 7

0 / 3 pts

Let's say your were going to use prime 2^{31}-1 as the modulus in a polynomial hash. One choice you must make in your design is in how many bits per chunk you will break the data you wish to hash. How many bits per chunk would you chose, and why?

Give two reasons for your choice and answer very briefly.

Your Answer:

~~128 bits~~

~~Allow bits of passes through permutation to scramble the bits~~

~~Slow hashing, will allow for more security of the hash, making it harder to reverse or guess~~

Rubric: 1 pt for picking < 31 bits, 1 pt for saying it fits in Z_p, 1 pt for picking a multiple of 8 and mentioning efficiency.

$p = 2^{31} - 1$ takes 31 bits

$Z_p = \{0, \ldots 2^{31} - 2\}$

biggest chunk guaranteed to work = 30 bits

24 is good, efficient

16 is good too, easy to read, more multiplications