**CPEG 572 Data and Computer Communications**

**ASSIGNMENT #7**

**Ch10:**

***Q.1***

What is the Hamming distance for each of the following code words?

1. *d* (10000, 00000)
   10000 – 00000 = 10000
   Distance = 1
2. *d* (10101, 10000)
   10101 – 10000 = 00101
   Distance = 2
3. *d* (00000, 11111)
   00000 – 11111 = 11111
   Distance = 5
4. *d* (00000, 00000)
   00000 – 00000 = 0
   Distance = 0

***Q.2***

Answer the following questions:

1. What is the polynomial representation of 101110?
   $x^5 + x^3 + x^2 + x$

2. What is the result of shifting 101110 three bits to the left?
   101110000
3. Repeat part b using polynomials.
   $x^8 + x^6 + x^5 + x^4$
4. What is the result of shifting 101110 four bits to the right?
   000010
5. Repeat part d using polynomials.
   x

## *Q.3*

Given the dataword 101001111 and the divisor 10111, show the generation of the CRC codeword at the sender site (using binary division).

```
         100110111
10111 | 1010011110000
         10111
          00111
          00000
           01111
           00000
            11111
            10111
             10001
             10111
              01100
              00000
               11000
               10111
                11110
                10111
                 10010
                 10111
                  0101 remainder
```

The codeword is 1010011110101

## *Q.4*

Assume we need to create codewords that can automatically correct a one-bit error. What should the number of redundant bits (r) be, given the number of bits in the dataword (k)? Remember that the codeword needs to be n = k + r bits, called C(n, k). After finding the relationship, find the number of bits in r if k is 1, 2, 5, 50, or 1000.
The redundant bits in this case need to find (n + 1) different states because the corruption can be in any of the n bits or zero bits. A set of r bits can define 2r states. This means that we need to have the following relationship $2^r \geq n + 1$. We need to solve the equation for each value of k

- if k = 1, r = 2, n = 3
  $2^2 \geq 3 + 1$, which means C(3, 1)
- if k = 2, r = 3, n = 5
  $2^3 \geq 5 + 1$, which means C(5, 2)

- if k = 5, r = 4, n = 9
  $2^4 \geq 9 + 1$, which means C(9, 5)
- if k = 50, r = 6, n = 56
  $2^6 \geq 56 + 1$, which means C(56, 50)
- if k = 1000, r = 10, n = 1010
  $2^{10} \geq 1010 + 1$, which means C(1010, 1000)

***<u>Q.5</u>***

Assume that the probability that a bit in a data unit is corrupted during transmission is p. Find the probability that x number of bits are corrupted in an nbit data unit for each of the following cases.

The equation to find the probability that x number of bits are corrupted in n bit data

$C(n,x)p^x(1-p)^{n-x}$

a. n = 8, x = 1, p = 0.2
   $C(8,1)\ 0.2\ (0.8)^7 = 0.34$
b. n = 16, x = 3, p = 0.3
   $C(16,3)0.3^3(0.7)^{13} = 0.15$
c. n = 32, x = 10, p = 0.4
   $C(32,10)0.4^{10}(0.6)^{22} = 0.09$

***<u>Q.6</u>***

Traditional checksum calculation needs to be done in one's complement arithmetic. Computers and calculators today are designed to do calculations in two's complement arithmetic. One way to calculate the traditional checksum is to add the numbers in two's complement arithmetic, find the quotient and remainder of dividing the result by $2^{16}$, and add the quotient and the remainder to get the sum in one's complement. The checksum can be found by subtracting the sum from $2^{16} - 1$. Use the above method to find the checksum of the following four numbers: 43,689, 64,463, 45,112, and 59,683

| **Decimal** | **Binary** | **2's complement** |
|---|---|---|
| 43689 | 1010101010101001 | 0101010101010l001 |
| 64463 | 1111101111001111 | 0111110111100111 |
| 45112 | 1011000000111000 | 0101100000011100 |
| 59683 | 1110100100100011 | 0111010010010001 |
| | **Sum** | 0110011111111010011 |

We will divide the sum with $2^{16}$

$2^{16}$ = 65536 = 1 0000 0000 0000 0000

Quotient: 011

Remainder: 011 1111 1101 0011

Quotient + Remainder = 011 1111 1101 0110
$2^{16}$ - 1 = 0 1111 1111 1111 1111
$2^{16}$ - (Quotient + Remainder) = 0 1100 0000 0010 1001= 49,193 as decimal

***Q.7***

An ISBN-13 code, a new version of ISBN-10, is another example of a weighted checksum with 13 digits, in which there are 12 decimal digits defining the book and the last digit is the checksum digit. The code, $D_1D_2D_3D_4D_5D_6D_7D_8D_9D_{10}D_{11}D_{12}C$, satisfies the following.

$$[(1 \cdot D\mathbf{1}) + (3 \cdot D\mathbf{2}) + (1 \cdot D\mathbf{3}) + \ldots + (3 \cdot D\mathbf{12}) + (1 \cdot \mathbf{C})] \bmod 10 = 0$$

We will calculate the sum of mod 10 of all digits. Then we let the check digit to be 10 – sum. So when the check digit is added to sum, the result is 0 mod 10.

$$[(1*9)+(3*7)+(1*8)+(3*0)+(1*0)+(3*7)+(1*2)+(3*9)+(1*6)+(3*7)+(1*7)+(3*5)]\ mod\ 10 = 7$$

Hence,

$$C = 10 - 7 = 3$$

***Q.8***

Assume we want to send a dataword of two bits using FEC based on the Hamming distance. Show how the following list of datawords/codewords can automatically correct up to a one-bit error in transmission.
Hamming distance principle “the code can correct one bit error d=2t+1”. The distance between dataword is 3. Assuming both sender/receiver has the list. The valid dataword to find one bit difference can be done by XOR the dataword with valid dataword.

00 → 00000 XOR 11011 = 11011
01 → 01011 XOR 11011 = 10000
10 → 10101 XOR 11011 = 01110
11 → 11110 XOR 11011 = 00101