Error Coding

Demo: https://youtu.be/pA6HpbJ3ntI

Modulo-2 multiplication and division

Implement the following Python code and complete Table 1.

http://www.asecuritysite.com/comms/mod2

Table 1: Multiply table

Value 1	Value 2	Modulo-2 multiplication
101010111	**101101**	
101010111	**110011**	
101010111	**100110**	
101010111	**111010**	
101010111	**100001**	
101010111	**100000**	
101010111	**100010**	

Prove these results using the following link:

http://www.ee.unb.ca/cgi-bin/tervo/calc.pl

Implement the following Python code and complete Table 2:

http://www.asecuritysite.com/comms/mod_div

Table 2: Division table

Value 1	Value 2	Modulo-2 result	Modulo-2 remainder
101010111	**101101**		
101010111	**110011**		
101010111	**100110**		
101010111	**111010**		
101010111	**100001**		
101010111	**100000**		
101010111	**100010**		

You have been asked to produce a sending Python program in which you have 8-bit

input values. In order to detect errors you have been asked to multiply the sending byte by “10101”.Now generate each possible value, and outline the first eight values:

```
0000 0000:
0000 0001:
0000 0010:
0000 0011:
0000 0100:
0000 0101:
0000 0110:
0000 0111:
```

A sample run with a range of 0 to 255 and with a multiplier of ‘11011’ is:

```
$ python ex05_01.py 255 11011
00000000        00000
00000001        11011
00000010        110110
00000011        101101
00000100        1101100
00000101        1110111
00000110        1011010
00000111        1000001
00001000        11011000
..
```

Now produce the receiving program, and input the sending values given above, and provide that the remainder is zero:

Value sent	**Value received**	**Value received correctly**
0000 0000		Y / N
0000 0001		Y / N
0000 0010		Y / N
0000 0011		Y / N
0000 0100		Y / N
0000 0101		Y / N
0000 0110		Y / N
0000 0111		Y / N

Now produce the receiving program, and input the sending values given above, and provide that the remainder is zero:

Value sent	Value received (with one error)	Error detected
0000 0000		Y / N
0000 0001		Y / N
0000 0010		Y / N
0000 0011		Y / N
0000 0100		Y / N
0000 0101		Y / N
0000 0110		Y / N
0000 0111		Y / N

Block parity

Block parity is a block code which adds a parity symbol to the end of a block of code. For example a typical method is to transmit the one's complement (or sometimes the two's complement) of the modulo-2 sum of the transmitted values. Implement a Python program which determines the block parity value for:

(i) {00000001, 00000100, 00001100, 11111111, 10011010, 00010001, 01000000, 01110110}
(ii) {00000111, 11100100, 00011100, 01111111, 10011010, 00010001, 01011000, 01111111}
(iii) { 00011111, 11111100, 10011100, 01111111, 10011010, 11010001, 01011000, 01111111}

A coding example is here:

http://asecuritysite.com/comms/block

For the following received values (with the last one being the received block parity check, determine if there is an error in the transmitted data, and, if so, where the error is:

(i) { 00011011, 11111100, 10011100, 01111111, 11111010, 00010001, 01011000, 01001111} Block parity value {11111000}
(ii) { 11011011, 11111100, 10011100, 01110011, 10110010, 00010001, 01011000, 01001111} Block parity value {01111000}

(iii) { 11011011, 11111100, 10011100, 01110011, 10110011, 10010001, 01011000, 01001100} Block parity value { 11111111}

Update the program so that it generates a 16-bit value checksum from eight 16-bit values. Outline how you have achieved this:

Update the program so that the it takes eight values and the parity block value, and identifies if there are errors and the position of the error (assuming a single bit in error). Outline how you have achieved this:

The program uses **even parity**, where the number of 1s is even. Update the program so that it also calculates the block code with odd parity. Outline how you have achieved this:

CRC

Using the following and the Python program outlined at:

http://asecuritysite.com/comms/crc_div

determine the transmitted bits for the following:

(i) $P(x) = x^3 + x^2 + x^0$ (1101)
$G(x) = x^6 + x^3 + x^2$ (1001100)

(ii) $P(x) = x^3 + x^2 + x^0$ (1101)
$G(x) = x^5 + x^3 + x^2$ (0101100)

Check your answer by using:

http://asecuritysite.com/comms/mod_div

and show there is a zero remainder in each case.

LRC

Implement an LRC/VRC calculator from:

http://asecuritysite.com/calculators/lrc

Use this program to complete the following table:

String	LRC	VRC
captain	0x01	0,0,0,1,1,1,0
jelly999		
Fishoil		
1234567		

Hamming distance

Implement a Hamming distance checker using the Python program defined on this page:

http://asecuritysite.com/calculators/ham

Use this program to complete the following table:

Value 1	Value 2	Hamming distance
0000 0000	**0101 1010**	
0000 0001	**0101 1010**	
0000 0010	**0101 1010**	
0000 0011	**0101 1010**	
0000 0100	**0101 1010**	
0000 0101	**0101 1010**	
0000 0110	**0101 1010**	

0000 0111	**0101 1010**	

With codes we might need to find the minimum Hamming distance for a range of codes. Implement the code here:

http://www.asecuritysite.com/comms/ham2

Now determine the minimum Hamming distance for the following codes:

- {0110,1111,0011,1010}
- {00001100,0110 0101, 1011 0011,11111111}

Note to install Commpy and bitstring you can use:

```
git clone https://github.com/veeresht/CommPy.git
cd CommPy
python setup.py install
pip install bitstring
```

Answers

1.5. A sample program is here:

https://dl.dropboxusercontent.com/u/40355863/ex03_01.zip