COM 5336 ASSIGNMENT #2

DUE BY 11:59PM **4/25**/2018 (Wed)

10% penalty applies to 1-day late submissions received between 12:00AM 4/26 and 11:59PM 4/26. No submission will be accepted after 12:00AM 4/27/2018

Objective

Implement the Advanced Encryption Standard (AES).

Description

First, implement GF(256) as follows. We use 8 bits to represent a polynomial of degree at most 7 as explained in the class. We can also use 8 bits to represent a monic polynomial of degree 8. For example, $m(x)=x^8+x^4+x^3+x+1$ can be represented as 0x1b. Represent GF(256) as F2[x]/m(x). Implement the following field operations as C/C++ functions:

```
uint8_t GF256_add(uint8_t a, uint8_tb, uint8_t mx);
// returns a + b. mx is the irreducible polynomial

uint8_t GF256_mult_x(uint8_t a, uint8_t mx);
// Multiplied by x. mx is the irreducible polynomial

uint8_t GF256_mult(uint8_t a, uint8_t b, uint8_t mx);
// General multiplication: mx is the irreducible polynomial

uint8_t GF256_inv(uint8_t *a, uint8_t mx);
// Returns the multiplicative inverse of a. mx is the irreducible polynomial
```

Implement AES as follows.

```
void AES_Encrypt(uint8_t* Plaintext, uint8_t* Ciphertext, uint8_t* Key);
void AES_Decrypt(uint8_t* Plaintext, uint8_t* Ciphertext, uint8_t* Key);
```

Write a main function that calls `AES_Encrypt()` and `AES_Decrypt()`. Show the (intermediate) state of each round as well as the final result in hex. Note that you MUST NOT use table look-ups for SubBytes. Compute the multiplicative inverse by calling GF256_inv(). In order to make your program I/O more readable, it is required to organize all data in hex numbers.

Grading

Your program MUST BE compatible with Dev C/C++ or GNU C/C++ compilers. If you are using other compilers, please make sure your final program is compatible. **You will get no points if your program is not compilable using the abovementioned compilers.** If your program is compilable but the result is not completely correct, you'll still get partial credits. Your program should be well-commented, well-structured, and easy to understand. You may lose up to 30% of points if you fail to do so. Each student is required to select a time slot and demo his/her program at EECS 625. Details of demo time will be announced in class on 4/11.

Submission

Put all your source codes in a folder containing main functions, function implementations, class definitions, or compilation instructions, if any. Compress them as a single zip file. DO NOT submit executable files. Name your zip file as your student ID number (i.e. 100012345.zip). Submit your source code on iLMS at http://lms.nthu.edu.tw.

Sample Input Subroutine Implementation

```
void input_func(uint8_t* input){
   uint8_t temp[49];
   int ch,i,c;
   int temp1[49];
   for(c=0;c<48;c++)
       temp[c]=getchar();                    //collect input(hex)
   for(c=0;c<48;c++){                        //transform to decimal
       if(temp[c] >= '0' && temp[c] <= '9')  temp1[c]=    temp[c]-'0';
       if(temp[c] >= 'a' && temp[c] <= 'f')  temp1[c]= 10+temp[c]-'a';
       if(temp[c] >= 'A' && temp[c] <= 'F')  temp1[c]= 10+temp[c]-'A';
       for(i=0;i<16;i++)
           input[i]=((16*temp1[3*i])+temp1[3*i+1]);     //transform to binary
   }
}
```

Sample I/O (Input shown in **blue**.)

Please show the result after each round of encryption and decryption. It is required to use the following format to show the result. The TAs will pick some elements to let you answer on the demo day.

```
<AES Encryption>
Plaintext: a3 c5 08 08 78 a4 ff d3 00 ff 36 36 28 5f 01 02
Key:       36 8a c0 f4 ed cf 76 a6 08 a3 b6 78 31 31 27 6e

--------Encryption--------
S0: 95 4f c8 fc 95 6b 89 75 08 5c 80 4e 19 6e 26 6c
S1: b8 8e 7b 5f d0 85 77 2d ba 81 47 eb 88 01 cc 99
S2: 69 66 3a 41 30 d9 3a c2 db 3e bc 8c d5 89 b1 8c
S3: a1 08 a2 2e d8 48 31 5d 3d 28 fd f6 c5 72 ad cb
S4: 5c c0 34 88 e9 da 78 b0 28 99 1f 10 20 02 37 ac
S5: e1 12 c2 ac 45 d2 2b c9 d0 53 3a 71 aa 5f a2 f2
S6: 94 00 6c e3 e7 91 46 9c 30 e0 eb 8c 4f bb 89 64
S7: 12 2a 3d 36 65 3d 98 bd 19 d1 de 38 2c c9 fd a1
S8: 57 97 d6 e9 c9 5e 0b 3e 68 c4 ff c9 c9 ce 25 59
S9: 0a 07 b8 81 6d 31 93 45 7b d2 cc 4b b9 fc 73 9f
Ciphertext: a6 24 62 48 34 dd a8 b9 1a f1 73 5d 00 0e cf 61

--------Decryption--------
S'0: 67 c7 4b db 3c b5 8f 0c 21 b0 6c 6e 56 c5 dc b3
S'1: 5b 58 16 cb dd 1c 3f 1e 45 8b f6 b2 dd 88 2b dd
S'2: c9 27 1d 32 4d 3e 54 05 d4 dd 27 7a 71 e5 46 07
S'3: 22 81 e9 43 94 e1 a7 11 04 ea 50 de 84 63 5a 64
S'4: f8 b5 80 89 6e ed 3a 91 70 cf 25 dd ac c9 f1 a3
S'5: 4a 57 c0 91 1e ee 9a c4 34 77 18 e7 b7 ba bc ca
S'6: 32 52 54 1f 61 34 95 31 27 40 3a 4c a6 30 c7 42
S'7: f9 35 65 64 04 b2 c8 83 b9 a7 80 25 03 33 80 64
S'8: 6c 97 a0 ee 70 0c 4b cf f4 7c 21 d8 c4 19 f5 e9
S'9: 2a 7f cd 50 2a 4a f7 b0 30 9f e8 9d d4 84 a7 2f
Plaintext: a3 c5 08 08 78 a4 ff d3 00 ff 36 36 28 5f 01 02
```

The example of test at demo (Answer shown in **blue**.):

S2 3	S5 9	S9 10	Ciphertext 16	S'0 4	S'4 5	S'6 1	S'8 12
3a	d0	d2	61	db	6e	32	d8

Note: S2-3 means the third element in round 2 of the encryption, and S'0-4 means the forth element in round 0 of the decryption.