**《应用密码学》实验指导书**

**实验一 AES变换操作的实现**

**一、实验目的**

（1）加深对AES算法的理解；

（2）阅读标准(fips-197)和文献，提高自学能力；

（3）加深对模块化设计的理解，提高编程实践能力。

**二、实验内容**

（1）按照AES算法，完成AES算法S盒、行移位、列混合、轮密钥加操作；

**三、实验要求**

（1）把AES的S盒、行移位、列混合、轮密钥加操作**写成一个函数**，然后再主程序中调用；（备注：函数名称命名方式 \*\*\*\*\_学号后2位，变量名称\*\*\*\_学号后2位，例如学号为“2015011263”S盒函数 SubBytes\_63(unsigned char input\_63[4][4]）

（2）输入128bits一个状态矩阵，输入为16进制数。状态矩阵的录入为一行，比如:

**32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34**

则轮密钥加的子密钥如：

**a0 fa fe 17 88 54 2c b1 23 a3 39 39 2a 6c 76 05**

(请注意这里是**列向优先**读取数据的)

（3）与**标准中的样例**进行比较,输入标准中样例，测试S盒、行移位、列混合和轮密钥加算法正确。

（4）实验程序测试中，二组测试结果

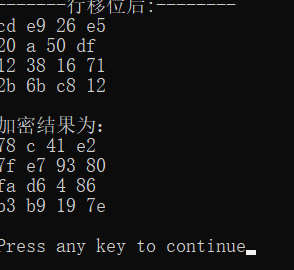
第1组：

状态矩阵的录入为一行，比如:

19 3d e3 be a0 f4 e2 2b 9a c6 8d 2a 9a c6 8d 2a e9 f8 48 08

则轮密钥加的子密钥如：

**a0 fa fe 17 88 54 2c b1 23 a3 39 39 2a 6c 76 05**

字节替代、行移位、列混合、轮密钥加的测试结果 

第2组 例如学号为“2015011263”

状态矩阵的录入为一行，比如:

**20 15 01 12 63 00 00 00 00 00 00 00 00 00 00 0E**

则轮密钥加的子密钥如：

**a0 fa fe 17 88 54 2c b1 23 a3 39 39 2a 6c 76 05**

字节替代、行移位、列混合、轮密钥加的测试结果

（5）程序的输出部分写在主程序中，可以是4行4列，也可以为一行。若为一行，输出也要求为**列向优先**。

（6）认真填写实验报告。

**四、参考资料**

（1） AES 的S盒

static unsigned char Sbox[16\*16]=

{// populate the Sbox matrix

/\* 0 1 2 3 4 5 6 7 8 9 a b c d e f \*/

/\*0\*/ 0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76,

/\*1\*/ 0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0, 0xad, 0xd4, 0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0,

/\*2\*/ 0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34, 0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15,

/\*3\*/ 0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a, 0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75,

/\*4\*/ 0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84,

/\*5\*/ 0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf,

/\*6\*/ 0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8,

/\*7\*/ 0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2,

/\*8\*/ 0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73,

/\*9\*/ 0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb,

/\*a\*/ 0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,

/\*b\*/ 0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae, 0x08,

/\*c\*/ 0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a,

/\*d\*/ 0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86, 0xc1, 0x1d, 0x9e,

/\*e\*/ 0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94, 0x9b, 0x1e, 0x87, 0xe9, 0xce, 0x55, 0x28, 0xdf,

/\*f\*/ 0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0, 0x54, 0xbb, 0x16};

字节替代

void SubBytes(unsigned char State[N][N])

{

int i,j;

char high;

char low;

for (i = 0; i < N; i ++)

{

for (j = 0; j < N; j ++)

{

low = State[i][j] & 0x0F; //取低4位

high = (State[i][j] >> 4)&0x0f; //取高4位

State[i][j] = Sbox[16\*high + low];

}

}

}

（2）行移位

void ShiftRows(unsigned char State[N][N])

{

int i,j,k;

int shiftnum = 0;

char tmp;

for (i = 0; i < N; i ++)

{

for (j = 0; j < shiftnum; j ++) //循环左移一次

{

tmp = State[i][0];

for (k = 0; k < N-1; k ++)

{

State[i][k] = State[i][k+1];

}

State[i][k] = tmp;

}

shiftnum ++; //移位次数+1

}

}

（3）轮密钥加

void AddRoundKey(unsigned char State[N][N], unsigned char RoundKey[N][N])

{

int i,j;

for (j = 0; j < N; j ++)

{

for (i = 0; i < N; i ++)

State[i][j] = State[i][j] ^ RoundKey[i][j];

}

}

（4）列混合相关的x乘

跟X乘相关的：

unsigned char Aes::gfmultby01(unsigned char b)

{

return b;

}

unsigned char Aes::gfmultby02(unsigned char b)

{

if (b < 0x80)

return (unsigned char)(int)(b <<1);

else

return (unsigned char)( (int)(b << 1) ^ (int)(0x1b) );

}

unsigned char Aes::gfmultby03(unsigned char b)

{

return (unsigned char) ( (int)gfmultby02(b) ^ (int)b );

}

for(j=0;j<4;j++)

{ output[0][j]= gfmultby02(input[0][j]))^ gfmultby03(input[1][j]))^ gfmultby01(input[2][j]))^ gfmultby01(input[3][j]);

}

**实验二 RSA模幂运算的实现**

**一、实验目的**

（1）加深对RSA算法的理解；

（2）加深对平方乘算法和模重复平方法的理解。

（3）加深对模块化设计的理解，提高编程实践能力。

**二、实验内容：**

（1）按照平方乘算法和模重复平方法，分别计算am mod n。。

**三、实验要求**

（1）要求把平方乘方法和模重复平方方法**写成一个函数**。（备注：函数名称命名方式 \*\*\*\*\_学号后2位，变量名称\*\*\*\_学号后2位，例如学号为“2015011263”，平方乘算法，LRFun\_63(int a\_63,int m\_63,int n\_63）

（2）在主程序中读入数据, 输入顺序为：a m n （注：a,m,n都是小于1000的。）

（3）使用平方乘方法**和**模重复平方法分别进行计算。

（4）输出计算的中间结果及最终运算结果。

例如：

请输入数字（a, m, n）:X X X

用平方乘算法的计算过程为：

i=0, …

i=1,…

…

结果为：

（5）认真填写实验报告。(实验报告必须有测试结果调试)

测试结果要求：

（1）1组固定的数组设置 ，模重复平方和平方程测试，a= 7 m=22 n=31

（2）模重复平方，1组数值随机设置，a,m,n都是小于1000

（3）平方乘算法，数组设置 底数a随机设置，m为学号后两位整数，n为学号后4位整数

**实验三 消息摘要函数SHA-1算法的数据填充和扩展**

**一、实验目的**

（1）通过阅读FIPS-180-2标准，加深对消息摘要函数SHA-1的理解；

（2）掌握消息摘要函数SHA-1最后分组的填充和数据扩展；

（3）提高编程实践能力。

**二、实验内容**

（1）按照消息摘要函数SHA-1算法的标准FIPS-180-2的要求，从文件中读取消息，然后对消息分组，并对最后一个分组进行填充，并通过数据扩充算法扩充到80个字。

**三、实验要求**

（1）输入为ASCII码，程序的默认输入为FIPS-180-2中示例的“abc”。

（2）输出填充后的最后一个分组中的W0, W1，W14 ，W15.然后数据扩充到80个字，然后输出W16, W79 （十六进制）。其中填充过程写成一个函数，数据扩充过程写成一个函数，

数据扩充中循环移位也可以写成一个函数

例如：

输出为：

W0 : 61626380

W1 :00000000

W14:00000000

W15 : 00000018

….

(3) 认真填写实验报告。

测试结果要求：

1. 输入为ASCII码，程序的默认输入为FIPS-180-2中示例的“abc”
2. 输入为ASCII码，程序的各自学号，例如“**2014062070**”