#include<iostream>

#include<bits/stdc++.h>

using namespace std;

int p10[] = {2,4,1,6,3,9,0,8,7,5};

int p8[] = {5,2,6,3,7,4,9,8};

int IP[] = {1,5,2,0,3,7,4,6};

int EP[] = {3,0,1,2,1,2,3,0};

int row[] = {0,3};

int col[] = {1,2};

int arr[2];

int num;

int s1[] = {1,0,3,2,3,2,1,0,0,2,1,3,3,1,3,2};

int s2[] = {0,1,2,3,2,0,1,3,3,0,1,0,2,1,0,3};

int bin[2];

int rem[2];

int p4[] = {1,3,2,0};

int p4\_res1[4];

int IP\_inv[] = {3,0,2,4,6,1,7,5};

int\* bin\_f;

int\* bin\_s;

int binary\_res1[4];

int convert\_to\_integer(int arr[]){

num = arr[0]\*2 + arr[1]\*1;

return num;

}

int\* convert\_to\_binary(int num1){

int p=0;

if(num1 == 0){

for(int i=0;i<2;i++){

rem[p] = 0;

p++;

}

}

else if(num1 == 1){

rem[0]= 0;

rem[1]= 1;

}

else if(num1 == 2){

rem[0] = 1;

rem[1] = 0;

}

else{

rem[0] = 1;

rem[1] = 1;

}

/\*if(num1!= 0){

rem[p] = num1%2;

num1 = num1/2;

p++;

}

else

rem[p] = num1;

int j=0;

for(int i=1;i>=0;i--){

bin[j] = rem [i];

j++;

}

for(int i=0;i<2;i++)

cout<<" "<<bin[i];

return bin; \*/

return rem;

}

int \* Substitution\_box(int res[]){

int Row1[2],Row2[2];

int Col1[2],Col2[2];

int l=0,m=0,n=0,k=0;

for(int i=0;i<8;i++){

if(i==0 || i==3){

Row1[k] = res[row[k]];

k++;

}

else if(i==1 || i==2){

Col1[l] = res[col[l]];

l++;

}

else if(i==4 || i==7){

Row2[m] = res[row[m]+4];

m++;

}

else{

Col2[n] = res[col[n]+4];

n++;

}

}

cout<<"Row1: ";

for(int i=0;i<2;i++)

cout<<Row1[i];

cout<<" Col1: ";

for(int i=0;i<2;i++)

cout<<Col1[i];

cout<<" Row2: ";

for(int i=0;i<2;i++)

cout<<Row2[i];

cout<<" Col2: ";

for(int i=0;i<2;i++)

cout<<Col2[i];

cout<<endl;

int a,b,c,d;

a = convert\_to\_integer(Row1);

b = convert\_to\_integer(Col1);

c = convert\_to\_integer(Row2);

d = convert\_to\_integer(Col2);

cout<<"A: "<<a <<" B: "<<b<<" C: "<<c<<" D: "<<d <<endl;

//Construction of Matrix S1 and S2

int s1\_mat[4][4];

int s2\_mat[4][4];

k=0;

for(int i=0;i<4;i++){

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

if(k<16){

s1\_mat[i][j] = s1[k];

s2\_mat[i][j] = s2[k];

k++;

}

}

}

cout<<"S1 Matrix: "<<endl;

for(int i=0;i<4;i++){

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

cout<<" "<<s1\_mat[i][j];

}

cout<<endl;

}

cout<<"S2 Matrix: "<<endl;

for(int i=0;i<4;i++){

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

cout<<" "<<s2\_mat[i][j];

}

cout<<endl;

}

int x = s1\_mat[a][b];

int y = s2\_mat[c][d];

cout<<"X: "<<x<<" Y: "<<y<<endl;

//Convert integer to binary

bin\_f = convert\_to\_binary(x);

for(int i=0;i<2;i++){

cout<<"Bin\_f: "<< bin\_f[i] <<endl;

binary\_res1[i] = bin\_f[i];

}

bin\_s = convert\_to\_binary(y);

for(int i=0;i<2;i++){

cout<<" Bin\_s: "<< bin\_s[i]<<endl;

binary\_res1[i+2] = bin\_s[i];

}

//Combining bin\_f and bin\_s

cout<<"Combine result: ";

for(int i=0;i<4;i++)

cout<<binary\_res1[i];

//Apply P4 on S-BOX result

cout<<" Applying P4: ";

for(int i=0;i<4;i++){

p4\_res1[i] = binary\_res1[p4[i]];

cout<<p4\_res1[i];

}

return p4\_res1;

}

int main(){

string pltxt;

cout<<"Enter the plain text:";

cin>>pltxt;

string key;

cout<<"Enter key: ";

cin>>key;

cout<<"Plain text is: "<<pltxt<<endl;

cout<<"Key is: "<<key<<endl;

unsigned char p\_ten[10];

//calculating p10

cout<<"P10 is: ";

for(int i=0;i<10;i++){

p\_ten[i] = key[p10[i]];

cout<<p\_ten[i];

}

cout<<endl;

//LS1

unsigned char temp1[5];

for(int j=4;j>=0;j--){

if(j==0){

temp1[4] = p\_ten[j];

}

else{

temp1[j-1] = p\_ten[j];

}

}

cout<<"LS1 first half is : ";

for(int i=0;i<5;i++)

cout<<temp1[i];

cout<<endl;

//LS1 next

unsigned char temp2[5];

int k=4;

for(int j=9;j>=5;j--){

if(j==5){

temp2[4] = p\_ten[j];

}

else{

temp2[k-1] = p\_ten[j];

}

k--;

}

cout<<"LS1 next half is : ";

for(int i=0;i<5;i++)

cout<<temp2[i];

cout<<endl;

//combine temp1 and temp2

unsigned char temp[10];

int j=0;

for(int i=0;i<10;i++){

if(i<5)

temp[i] = temp1[i];

else{

temp[i] = temp2[j];

j++;

}

}

cout<<"After combining two arrays(LS1): ";

for(int i=0;i<10;i++)

cout<<temp[i];

cout<<endl;

// Now applying P8 on LS1

unsigned char key1[8];

for(int i=0;i<8;i++)

key1[i] = temp[p8[i]];

cout<<"After applying p8: " <<endl<<"Key1 is: ";

for(int i=0;i<8;i++)

cout<<key1[i];

cout<<endl;

unsigned char temp3[5];

unsigned char temp4[5]; //Obtaining LS2 first 5-bits

for(int i=4;i>=0;i--){

if(i>1){

temp3[i-2] = temp1[i];

}

if(i==0)

temp3[3] = temp1[i];

else

temp3[4] = temp1[i];

}

cout<<"First\_half of LS2 is: ";

for(int i=0;i<5;i++)

cout<<temp3[i];

cout<<endl;

//obtaining LS2 next 5-bits

for(int i=4;i>=0;i--){

if(i>1){

temp4[i-2] = temp2[i];

}

if(i==0)

temp4[3] = temp2[i];

else

temp4[4] = temp2[i];

}

cout<<"Second\_half of LS2 is: ";

for(int i=0;i<5;i++)

cout<<temp4[i];

cout<<endl;

//combining LS2(10-bits)

unsigned char ttemp[10];

int p=0;

for(int i=0;i<10;i++){

if(i<5)

ttemp[i] = temp3[i];

else{

ttemp[i] = temp4[p];

p++;

}

}

cout<<"Combined LS2 is: ";

for(int i=0;i<10;i++)

cout<<ttemp[i];

cout<<endl;

//Applying p8 on LS2

unsigned char key2[8];

for(int i=0;i<8;i++)

key2[i] = ttemp[p8[i]];

cout<<"After applying p8 on LS2: ";

for(int i=0;i<8;i++)

cout<<key2[i];

cout<<endl;

unsigned char Ip\_pt[8]; //Applying IP to plain text

cout<<"After applying IP to plain text: ";

for(int i=0;i<8;i++){

Ip\_pt[i] = pltxt[IP[i]];

cout<<Ip\_pt[i];

}

cout<<endl;

//Divide this array for calculating EP

unsigned char ip\_first\_half[4];

unsigned char ip\_second\_half[4];

for(int i=0;i<8;i++){

if(i<4)

ip\_first\_half[i] = Ip\_pt[i];

else

ip\_second\_half[i-4] = Ip\_pt[i];

}

cout<<"Ip first half: ";

for(int i=0;i<4;i++)

cout<<ip\_first\_half[i];

cout<<endl;

cout<<"Ip second half: ";

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

cout<<ip\_second\_half[j];

cout<<endl;

unsigned char Ep\_array[8]; //calculate EP

for(int j=0;j<8;j++)

Ep\_array[j] = ip\_second\_half[EP[j]];

cout<<"EP is: ";

for(int k=0;k<8;k++)

cout<<Ep\_array[k];

cout<<endl;

// Perform X-OR operation on EP and key1

int x\_or\_res1[8];

cout<<"X-OR result of EP and Key1: ";

for(int k=0;k<8;k++)

x\_or\_res1[k] = Ep\_array[k] ^ key1[k];

for(int k=0;k<8;k++)

cout<<x\_or\_res1[k];

cout<<endl;

//Substitution Box

int\* p4\_sub\_res;

p4\_sub\_res = Substitution\_box(x\_or\_res1);

cout<<" p4 apply on s-box result is: ";

for(int i=0;i<4;i++)

cout<< p4\_sub\_res[i];

//Take X-OR of P4 and First\_half of IP

int x\_or\_res2[4];

cout<<" X-OR Result of p4 and first half of IP: ";

for(int i=0;i<4;i++)

x\_or\_res2[i] = (p4\_sub\_res[i] ^ ip\_first\_half[i]) - 48;

for(int i=0;i<4;i++){

cout<<x\_or\_res2[i];

}

//Replacing first 4-bit of IP result with XOR result

for(int i=0;i<8;i++){

if(i<4)

Ip\_pt[i] = x\_or\_res2[i];

else

Ip\_pt[i] = ip\_second\_half[i-4];

}

//perform swapping

int Ip\_pt\_swap[8];

for(int i=0;i<8;i++){

if(i<4)

Ip\_pt\_swap[i] = (Ip\_pt[i+4] - 48);

else

Ip\_pt\_swap[i] = Ip\_pt[i-4];

}

cout<<" Swap result: ";

for(int j=0;j<8;j++)

cout<<Ip\_pt\_swap[j];

//Apply EP on next 4-bits of Ip\_pt\_swap

int Ep\_array1[8];

cout<<endl<<"EP array 1: ";

for(int i=0;i<8;i++){

Ep\_array1[i] = x\_or\_res2[EP[i]];

cout<< Ep\_array1[i];

}

//Apply X-OR operation on E/p and Key2

int x\_or\_res3[8];

cout<<endl<<"XOR Result with key2: ";

for(int k=0;k<8;k++){

x\_or\_res3[k] = (Ep\_array1[k] ^ key2[k]) -48;

cout<<x\_or\_res3[k];

}

//Calling to Substution box function

int\* p4\_sub\_res1;

p4\_sub\_res1 = Substitution\_box(x\_or\_res3);

cout<<endl<<"p4\_sub\_res is: ";

for(int i=0;i<4;i++)

cout<< p4\_sub\_res1[i];

//Take XOR on P4 and 4-bits of ip\_pt\_swap

int x\_or\_res4[4];

cout<<endl<<"XOR result p4 and 4-bits of ip\_pt\_swap: ";

for(int k=0;k<4;k++){

x\_or\_res4[k] = p4\_sub\_res1[k] ^ Ip\_pt\_swap[k];

cout<<x\_or\_res4[k];

}

//Swap first 4-bit of XOR result with ip\_pt\_swap

int Ip\_pt\_swap2[8];

for(int i=0;i<8;i++){

if(i<4)

Ip\_pt\_swap2[i] = x\_or\_res4[i];

else

Ip\_pt\_swap2[i] = Ip\_pt\_swap[i];

}

cout<<endl<<"Ip\_pt\_swap2 result is: ";

for(int i=0;i<8;i++)

cout<<Ip\_pt\_swap2[i];

int cipher\_text[8]; //Apply IP-inverse on ip\_pt\_swap2 to obtain cipher text

for(int i=0;i<8;i++)

cipher\_text[i] = Ip\_pt\_swap2[IP\_inv[i]];

cout<<endl<<"cipher text is: ";

for(int k=0;k<8;k++)

cout<<cipher\_text[k];

// Decryption Procedure

//Apply IP to cipher text

int ip\_cipher\_txt[8];

for(int i=0;i<8;i++)

ip\_cipher\_txt[i] = cipher\_text[IP[i]];

cout<<" ip\_cipher\_txt: ";

for(int i=0;i<8;i++)

cout<<ip\_cipher\_txt[i];

//Apply EP on last 4-bits

int ip\_cipher\_next\_half[4];

for(int k=0;k<4;k++)

ip\_cipher\_next\_half[k] = ip\_cipher\_txt[k+4];

int Ep\_ip\_cipher\_txt[8];

for(int j=0;j<8;j++)

Ep\_ip\_cipher\_txt[j] = ip\_cipher\_next\_half[EP[j]];

cout<<endl<<"Ep result: ";

for(int m=0;m<8;m++)

cout<<Ep\_ip\_cipher\_txt[m];

//Take XOR with key2

int x\_or\_res\_decrypt1[8];

cout<<endl<<" XOR result : " ;

for(int i=0;i<8;i++){

x\_or\_res\_decrypt1[i] = (Ep\_ip\_cipher\_txt[i] ^ key2[i]) - 48 ;

cout<<x\_or\_res\_decrypt1[i];

}

//Apply Substitution box

int\* p4\_sub\_decrypt1;

p4\_sub\_decrypt1 = Substitution\_box(x\_or\_res\_decrypt1);

cout<<endl<<"P4 in decryption: ";

for(int n=0;n<4;n++)

cout<<p4\_sub\_decrypt1[n];

//Take XOR of p4 and ip first 4 bits

int xor\_dec\_res1[4];

cout<<endl<<"XOR result: ";

for(int i=0;i<4;i++){

xor\_dec\_res1[i] = p4\_sub\_decrypt1[i] ^ ip\_cipher\_txt[i];

cout<<xor\_dec\_res1[i];

}

int swap\_dec\_res1[8];

for(int i=0;i<8;i++){

if(i<4)

swap\_dec\_res1[i] = ip\_cipher\_txt[i+4];

else

swap\_dec\_res1[i] = xor\_dec\_res1[i-4];

}

cout<<endl<<"After swapping we get: ";

for(int c=0;c<8;c++)

cout<<swap\_dec\_res1[c];

int Ep\_xor\_dec\_res1[8]; //Apply EP on xor\_dec\_res1

for(int u=0;u<8;u++)

Ep\_xor\_dec\_res1[u] = xor\_dec\_res1[EP[u]];

//Take XOR of EP and key1

int Ep\_xor\_key1\_res[8];

cout<<endl<<"Xor result with key1: ";

for(int i=0;i<8;i++){

Ep\_xor\_key1\_res[i] = (Ep\_xor\_dec\_res1[i] ^ key1[i]) - 48;

cout<< Ep\_xor\_key1\_res[i];

}

//call substitution box

int\* p4\_sub\_decrypt2;

p4\_sub\_decrypt2 = Substitution\_box(Ep\_xor\_key1\_res);

cout<<endl<<"p4\_sub\_decrypt2: ";

for(int i=0;i<4;i++)

cout<< p4\_sub\_decrypt2[i];

//Take XOR of p4 and first 4-bit of swap\_dec\_res1

int xor\_dec\_res2[4];

cout<<" Xor output: ";

for(int i=0;i<4;i++){

xor\_dec\_res2[i] = (p4\_sub\_decrypt2[i] ^ swap\_dec\_res1[i]);

cout<<xor\_dec\_res2[i];

}

//replace first 4-bit of swap\_dec\_res1 with xor result

int swap\_dec\_res2[8];

for(int i=0;i<8;i++){

if(i<4)

swap\_dec\_res2[i] = xor\_dec\_res2[i];

else

swap\_dec\_res2[i] = swap\_dec\_res1[i];

}

int decryption\_res[8];

cout<<endl<<"Final Decryption result: "; //Take IP-inverse for final result

for(int i=0;i<8;i++){

decryption\_res[i] = swap\_dec\_res2[IP\_inv[i]];

cout<<decryption\_res[i];

}

cout<<endl;

return 0;

}

/\*Output is:

guest-i9K7Ws@admin13-HP-Compaq-4000-Pro-SFF-PC:~$ g++ a.cpp

guest-i9K7Ws@admin13-HP-Compaq-4000-Pro-SFF-PC:~$ ./a.out

Enter the plain text:10100101

Enter key: 0010010111

Plain text is: 10100101

Key is: 0010010111

P10 is: 1000010111

LS1 first half is : 00001

LS1 next half is : 01111

After combining two arrays(LS1): 0000101111

After applying p8:

Key1 is: 00101111

First\_half of LS2 is: 00100

Second\_half of LS2 is: 11101

Combined LS2 is: 0010011101

After applying p8 on LS2: 11101010

After applying IP to plain text: 01110100

Ip first half: 0111

Ip second half: 0100

EP is: 00101000

X-OR result of EP and Key1: 00000111

Row1: 00 Col1: 00 Row2: 01 Col2: 11

A: 0 B: 0 C: 1 D: 3

S1 Matrix:

1 0 3 2

3 2 1 0

0 2 1 3

3 1 3 2

S2 Matrix:

0 1 2 3

2 0 1 3

3 0 1 0

2 1 0 3

X: 1 Y: 3

Bin\_f: 0

Bin\_f: 1

Bin\_s: 1

Bin\_s: 1

Combine result: 0111 Applying P4: 1110 p4 apply on s-box result is: 1110 X-OR Result of p4 and first half of IP: 1001 Swap result: 01001001

EP array 1: 11000011

XOR Result with key2: 00101001Row1: 00 Col1: 01 Row2: 11 Col2: 00

A: 0 B: 1 C: 3 D: 0

S1 Matrix:

1 0 3 2

3 2 1 0

0 2 1 3

3 1 3 2

S2 Matrix:

0 1 2 3

2 0 1 3

3 0 1 0

2 1 0 3

X: 0 Y: 2

Bin\_f: 0

Bin\_f: 0

Bin\_s: 1

Bin\_s: 0

Combine result: 0010 Applying P4: 0010

p4\_sub\_res is: 0010

XOR result p4 and 4-bits of ip\_pt\_swap: 0110

Ip\_pt\_swap2 result is: 01101001

cipher text is: 00110110 ip\_cipher\_txt: 01101001

Ep result: 11000011

XOR result : 00101001Row1: 00 Col1: 01 Row2: 11 Col2: 00

A: 0 B: 1 C: 3 D: 0

S1 Matrix:

1 0 3 2

3 2 1 0

0 2 1 3

3 1 3 2

S2 Matrix:

0 1 2 3

2 0 1 3

3 0 1 0

2 1 0 3

X: 0 Y: 2

Bin\_f: 0

Bin\_f: 0

Bin\_s: 1

Bin\_s: 0

Combine result: 0010 Applying P4: 0010

P4 in decryption: 0010

XOR result: 0100

After swapping we get: 10010100

Xor result with key1: 00000111Row1: 00 Col1: 00 Row2: 01 Col2: 11

A: 0 B: 0 C: 1 D: 3

S1 Matrix:

1 0 3 2

3 2 1 0

0 2 1 3

3 1 3 2

S2 Matrix:

0 1 2 3

2 0 1 3

3 0 1 0

2 1 0 3

X: 1 Y: 3

Bin\_f: 0

Bin\_f: 1

Bin\_s: 1

Bin\_s: 1

Combine result: 0111 Applying P4: 1110

p4\_sub\_decrypt2: 1110 Xor output: 0111

Final Decryption result: 10100101

\*/