LAB-1

```
#include <stdio.h>
#include <stdlib.h>
#include<ctype.h>
#include<string.h>

char* encrypt(char pt[30],char keymat[5][5])
{
   int i,j,k,w,x,y,z,c=0;
   char* cipher;
   char ch1,ch2;

   for(k=0;k<strlen(pt);k+=2)
   {
      ch1=pt[k];
      ch2=pt[k+1];
   for(i=0;i<5;i++)
   {
      for(j=0;j<5;j++)
      {

         if(ch1==keymat[i][j])
         {
            w=i;
            x=j;
         }
         else if(ch2==keymat[i][j])
         {
            y=i;
            z=j;
         }
      }
   }

   if(w==y)
   {
      x=(x+1)%5;
      z=(z+1)%5;

      cipher[c++]=keymat[w][x];
      cipher[c++]=keymat[y][z];
   }
   else if(x==z)
   {
      w=(w+1)%5;
      y=(y+1)%5;

       cipher[c++]=keymat[w][x];
      cipher[c++]=keymat[y][z];
   }
   else
   {
      cipher[c++]=keymat[w][z];
      cipher[c++]=keymat[y][x];
   }
```

```
 }
 cipher[c]='\0';
printf("\n");
return cipher;
}


char* decrypt(char cipher[20],char mat[5][5])
{
   char* plain;
   int plen=0,i,k,l;
   int c11,c12,c21,c22;
   char ch1,ch2;
   for(i=0;i<strlen(cipher);i=i+2)
   {
      ch1=cipher[i];
      ch2=cipher[i+1];
      for(k=0;k<5;k++)
      {
         for(l=0;l<5;l++)
         {
            if(ch1==mat[k][l])
            {
               c11=k;
               c12=l;
            }
            if(ch2==mat[k][l])
            {
            c21=k;
            c22=l;
            }
         }
      }
      if(c11==c21)
      {
         plain[plen++]=mat[c11][(c12-1)>0?(c12-1)%5:((c12+4)%5)];
          plain[plen++]=mat[c21][(c22-1)>0?(c22-1)%5:((c22+4)%5)];
      }
      else if(c12==c22)
      {
         plain[plen++]=mat[(c11-1)>0?(c11-1)%5:((c11+4)%5)][c12];
         plain[plen++]=mat[(c21-1)>0?(c21-1)%5:((c21+4)%5)][c22];
      }
      else
      {
         plain[plen++]=mat[c11][c22];
         plain[plen++]=mat[c21][c12];
      }
   }
   plain[plen]='\0';
   printf("plain text :\n");

   return plain;
}

void modify_ip(char input[20],char modify[20])
{
```

```
    int len=0,i=0;
    while(input[i]!='\0')
    {
        modify[len++]=input[i++];
        if(input[i]=='\0' || input[i]==input[i-1])
        {
            modify[len++]='X';
        }
        else
        modify[len++]=input[i++];
    }
    modify[len]='\0';
    printf("\n Modified plain text %s\n",modify);
}

int main()
{
    int i,j,m=0,k=0;
    char pt[30],key[30],keymat[5][5],keyminus[30],modify[30];
    char alpha[26]={'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'};
    char* final;
    char *cipher;
    printf("enter the plain text: ");
    gets(pt);
    printf("enter key: ");
    gets(key);

    for(i = 0; i < strlen(key); i++)
      {
         for(j = i + 1; key[j] != '\0'; j++)
         {
            if(key[j] == key[i])
          {
               for(k = j; key[k] != '\0'; k++)
             {
                key[k] = key[k + 1];
             }
          }
        }
    }


    for(i=0;i<strlen(pt);i++)
    {
        if(pt[i]=='j')pt[i]='i';
        else if(pt[i]=='J')pt[i]=='I';
        pt[i]=toupper(pt[i]);
    }

    for(i=0;i<strlen(key);i++)
    {
        if(key[i]=='j')key[i]='i';
        else if(key[i]=='J')key[i]=='I';
        key[i]=toupper(key[i]);
    }

    j=0;
```

```
    for(i=0;i<26;i++)
    {
       for(k=0;k<strlen(key);k++)
       {
          if(key[k]==alpha[i])
          break;
          else if(alpha[i]=='J')break;
       }
       if(k==strlen(key))
       {
          keyminus[j]=alpha[i];
          j++;
       }
    }

    k=0;
    printf("key table\n");
    for(i=0;i<5;i++)
    {
       for(j=0;j<5;j++)
       {
          if(k<strlen(key))
          {
          keymat[i][j]=key[k];
          k++;
          }
          else
          {
             keymat[i][j]=keyminus[m];
             m++;
          }
          printf("%c ",keymat[i][j]);
       }
       printf("\n");
    }
     modify_ip(pt,modify);
     cipher=encrypt(modify,keymat);
      printf("cipher text:%s",cipher);
     printf("\n");
     final=decrypt(cipher,keymat);

     for(i=0;i<strlen(final);i++)
     {
        if(final[i]!='X')
        printf("%c",final[i]);
     }
 return 0;
}
```

LAB-2

```
#include<stdio.h>

int euclid(int a, int m)
{
   int m0 = m;
   int y = 0, x = 1;

   if (m == 1)
     return 0;

   while (a > 1)
   {
      // q is quotient
      int q = a / m;
      int t = m;

      // m is remainder now, process same as
      // Euclid's algo
      m = a % m, a = t;
      t = y;

      // Update y and x
      y = x - q * y;
      x = t;
   }

   // Make x positive
   if (x < 0)
      x += m0;

   return x;
}

char * encrypt(char input[],int key[3][3],int dim)
{
int i,j,k,clen, len, flag=0,l;

int out[3][1], med[3][1];
char inp[30];
static char cipher[50];
for(i=0,len=0;i<strlen(input);i++)
if(input[i]!=' ' && isalpha(input[i]))
inp[len++]=tolower(input[i]);
inp[len]='\0';
i=len=strlen(inp);
while(len%dim!=0)
{
inp[i++]='x';
len++;
flag=1;
}
if(flag)
inp[len]='\0';
for(i=0,clen=0;i<strlen(inp);i=i+dim)
{
```

```
for(k=0;k<dim;k++)
med[k][0]=inp[i+k]-'a';
for(k=0;k<dim;k++)
{
out[k][0]=0;
for(l=0;l<dim;l++)
out[k][0]+=key[k][l]*med[l][0];
}
for(k=0;k<dim;k++)
cipher[clen++]=(out[k][0]%26)+'a';
}
cipher[clen]='\0';
return cipher;
}

void inverse(int key[3][3],int dim,int inv[3][3])
{
    int i,j,mat[3][3], a[5][5] ;
   int det = 0;
   if(dim==2)
   {
    det=key[0][0]*key[1][1]- key[0][1]*key[1][0];
   mat[0][0]=key[1][1];
   mat[0][1]=key[0][1]*-1;
   mat[1][0]=key[1][0]*-1;
   mat[1][1]=key[0][0];

   }
else
{
   //convert 3*3 to 5*5
  for(i=0;i<5;i++)
  {
     for(j=0;j<5;j++)
     {
        a[i][j]=key[i%3][j%3];
     }
  }

  for(i=0;i<3;i++)
  for(j=0;j<3;j++)
  {
     mat[j][i]=a[i+1][j+1]*a[i+2][j+2]-a[i+1][j+2]*a[i+2][j+1];
     if(mat[j][i]<0)
     mat[j][i]=-(25*mat[j][i])%26;  //negative mod
     else
     mat[j][i]=mat[j][i]%26;
  }

for(i = 0; i < 3; i++)
      det = det + (key[0][i] * (key[1][(i+1)%3] * key[2][(i+2)%3] - key[1][(i+2)%3] *key[2][(i+1)%3]));
}


if(det<0)
det=-(25*det)%26;
int mi; //multiplicative inverse
```

```
mi=euclid(det,26);

for(i=0;i<dim;i++)
for(j=0;j<dim;j++)
{
inv[i][j]=(mi*mat[i][j]);
if(inv[i][j]<0) //negative mod
inv[i][j]=-(25*inv[i][j])%26;
else
inv[i][j]=inv[i][j]%26;
}
}

char* decrypt(int key[3][3],int dim,char cipher[])
{
int i,j,k,l,len,temp=0;
int out[3][1],med[3][1];
int inv[3][3];
static char plain[50];
inverse(key,dim,inv);
for(len=0,i=0;i<strlen(cipher);i+=dim)
{
for(k=0;k<dim;k++)
med[k][0]=cipher[i+k]-'a';
for(k=0;k<dim;k++)
{
out[k][0]=0;
for(l=0;l<dim;l++)
out[k][0]+=inv[k][l]*med[l][0];
}
for(k=0;k<dim;k++)
plain[len++]=(out[k][0]%26)+'a';
}
plain[len]='\0';
return plain;
}



int main(){

    int key[3][3],dim;
char input[30], *cipher, *plain;
printf("Enter the plain text:");
scanf("%[^\n]s",input);
printf("Enter dimension:");
scanf("%d",&dim);
printf("Enter matrix:\n");
int i,j;
for(i=0;i<dim;i++)
for(j=0;j<dim;j++)
scanf("%d",&key[i][j]);
cipher=encrypt(input,key,dim);
printf("cipher:%s\n",cipher);

plain=decrypt(key,dim,cipher);
```

```
printf("text after decryption:\n");
for(i=0;i<strlen(plain);i++)
if(plain[i]!='x')
printf("%c",plain[i]);


  return 0;
}
```

LAB-3

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <time.h>
void decrypt(char key[]);

int main()
{
   char key[27];
   int i,j=0,arr[26]={0};
   srand(time(NULL));
   while(j<26){
      i=rand()%26;
      if(arr[i]==0){
         key[j++]='a'+i;
         arr[i]=1;
      }
   }
   key[j]='\0';
   int frequency[26]={0};
   FILE *f1,*f2;
   char text;
   f1=fopen("plaintext.txt","r");
   f2=fopen("ciphertext.txt","w");
   while(fscanf(f1,"%c",&text)!=EOF){
      if(isalpha(text)){
         frequency[text-'a']++;
         fprintf(f2,"%c",key[text-'a']);
      }
      else
         fprintf(f2,"%c",text);
   }
   fclose(f1);
   fclose(f2);
   printf("%s\n",key);

   for(i=0;i<26;i++)
      printf("|%d| %c -> %c\n",frequency[i],'a'+i,key[i]);
      decrypt(key);
   return 0;
}
```

```
void decrypt(char key[])
{
   int i;
   char text;
   FILE *f1,*f2;
   f1=fopen("decryptedtext.txt","w");
    f2=fopen("ciphertext.txt","r");
     while(fscanf(f2,"%c",&text)!=EOF){
       if(isalpha(text)){
          for(i=0;i<26;i++)
             if(key[i]==text)
               break;
          fprintf(f1,"%c",'a'+i);
       }
       else
          fprintf(f1,"%c",text);
   }
   fclose(f1);
   fclose(f2);
}
```

LAB-4

```
#include<stdio.h>
#include<string.h>


char *encrypt(char *p,int sequence[],int l1)
{
   int i,j,l2,order[l1];
   for(i=1;i<=l1;++i)
   {
      for(j=0;j<l1;++j)
      {
         if(sequence[j]==i)
            order[i-1]=j;
      }
   }
   l2=strlen(p);
   if(l2%l1!=0)
   {
      while(l2%l1!=0)
         p[l2++]='x';
      p[l2]='\0';
      printf("bogus char used:%c\n",'x');
      printf("final ip:%s",p);
   }
   int r=l2/l1;
   char p1[r][l1];
   int count=0,k=1;
   printf("\n encryption\n");


   count=0;

   k=k+1;
```

```
    for(i=0;i<r;i++)
    {
    for(j=0;j<l1;j++)
    {
    p1[i][j]=p[count];
    count=count+1;

    }
    }
    for(i=0;i<r;i++)
    {
       for(j=0;j<l1;j++)
       {
          printf("%c",p1[i][j]);
       }
       printf("\n");
    }
    count=0;
    for(i=0;i<l1;++i)
    {
       for(j=0;j<r;++j)
       {
          p[count]=p1[j][order[i]];
          count=count+1;
       }
    }


return(p);

}

char* decrypt(char *p,int sequence[],int l1)
{

    int i,j,r,l2;
    l2=strlen(p);
    r=l2/l1;
    char p1[20][20];
    int order[l1];
    for(i=1;i<=l1;++i)
    {
       for(j=0;j<l1;++j)
       {
          if(sequence[j]==i)
             order[i-1]=j;
       }
    }
    printf("decryption\n");
       int count=0;
       for(i=0;i<l1;i++)
       {
          for(j=0;j<r;j++)
          {
             p1[j][order[i]]=p[count];
             count=count+1;
          }
```

```
        }
    for(i=0;i<r;i++)
    {
        for(j=0;j<l1;j++)
        {
            printf("%c",p1[i][j]);
        }
        printf("\n");
    }
    count=0;
    for(i=0;i<r;i++)
    {
        for(j=0;j<l1;j++)
        {
            p[count]=p1[i][j];
            count=count+1;
        }

    }

return p;
}



int main()
{
    int l1,i;
    char pt[100];
    char* cipher,*Dpt;
    printf("\n enter length of key:\n");
    scanf("%d",&l1);
    int sequence[l1];
    printf("enter seq ky:\n");
    for(i=0;i<l1;++i)
    {
        scanf("%d",&sequence[i]);
    }


    printf("enter plain text without spaces\n");
    scanf("%s",pt);
    cipher=encrypt(pt,sequence,l1);
    printf("\n ciphertext:\n");
    for(i=0;i<strlen(cipher);i++)
        printf("%c",cipher[i]);
    printf("\n\n");
    Dpt=decrypt(cipher,sequence,l1);
    printf("after decryption plain text is:");
    for(i=0;i<strlen(Dpt);i++)
    if(Dpt[i]!='x')
    printf("%c",Dpt[i]);

return(0);
}
```

LAB-5

```
#include<stdio.h>
#include<math.h>
int pc1_key[56],c[28],d[28],keyshift[56],pc2_key[48];

void converttobinary(int hex[16],int bin[64])
{
int i;

for(i=0;i<64;i++)
bin[i]=0;

long int rem,count;
for(i=0;i<16;i++){
count=1;
 while (hex[i]>0)
{
rem=hex[i]%2;
bin[(i+1)*4-count]=rem;

hex[i]=hex[i]/2;
count++;
}
}
}



int pc1[8][7]={57,49,41,33,25,17,9,
                                    1,58,50,42,34,26,18,
                                    10,2,59,51,43,35,27,
                                    19,11,3,60,52,44,36,
                                    63,55,47,39,31,23,15,
                                    7,62,54,46,38,30,22,
                                    14,6,61,53,45,37,29,
                                    21,13,5,28,20,12,4
                                    };
int pc2[8][6]={
14,17,11,24,1,5,
3,28,15,6,21,10,
23,19,12,4,26,8,
16,7,27,20,13,2,
41,52,31,37,47,55,
30,40,51,45,33,48,
44,49,39,56,34,53,
46,42,50,36,39,32
};
```

```
void PC1(int key[64])
{
int i,j,k=0;
for(i=0;i<8;i++)
{
for(j=0;j<7;j++)
{
pc1_key[k++]=key[pc1[i][j]-1];
}
}
}



void leftshift()
{
int i,j,k=0;
k=c[0];
for(i=1;i<28;i++)
c[i-1]=c[i];
c[i-1]=k;
k=d[0];
for(i=1;i<28;i++)
d[i-1]=d[i];
d[i-1]=k;
for(i=0;i<28;i++)
keyshift[i]=c[i];
for(k=0,j=i;j<56;j++)
keyshift[i++]=d[k++];
}



void PC2()
{
int i,j,k=0;
for(i=0;i<8;i++)
{
for(j=0;j<6;j++)
{
pc2_key[k++]=keyshift[pc2[i][j]-1];
}
}
}



int main()
{
   FILE *myfile,*myfile1;
   myfile1=fopen("input.txt","r");
   myfile=fopen("output.txt","w");
int i,j,k=0,round=1,n,bin[64],hex[64];
printf("key entered in input file in hex format(0x0)");
for(i=0;i<16;i++)
fscanf(myfile1,"%x",&hex[i]);
```

```
converttobinary(hex,bin);


printf("\n 64 bit input is\n");
for(i=1;i<=64;i++)
{
   printf("%d",bin[i-1]);
   if(i%8==0)
   printf("\n");
}

printf("\n enter no of rounds\n");
scanf("%d",&n);
PC1(bin);
for(i=0;i<28;i++)
c[i]=pc1_key[i];
for(j=i;j<56;j++)
d[k++]=pc1_key[j];

while(round<=n)
{
if(round==1||round==2||round==9||round==16)
leftshift();
else
{
leftshift();
leftshift();
}
PC2();
printf("\n key for round %d:\n",round);

for(i=1;i<=48;i++)
{
printf("%d",pc2_key[i-1]);if(i%8==0)printf(" ");
if(myfile==NULL)
{
   printf("file not e\opened\n");
   return 1;
}

fprintf(myfile,"%d",pc2_key[i-1]);if(i%48==0)fprintf(myfile,"\n");
}

round++;
}

fclose(myfile);
return 0;
}
```

LAB-6

```
#include<stdio.h>
#include<math.h>
int l[32],r[32],er[48];
int pc2_key[48];
 int pt[64];
 int e_bit[8][6]={
            32,1,2,3,4,5,
            4,5,6,7,8,9,
           8,9,10,11,12,13,
           12,13,14,15,16,17,
           16,17,18,19,20,21,
           20,21,22,23,24,25,
           24,25,26,27,28,29,
           28,29,30,31,32,1
           };

int ip[8][8]={
              58,50,42,34,26,18,10,2,
              60,52,44,36,28,20.12,4,
              62,54,46,38,30,22,14,6,
              64,56,48,40,32,24,16,8,
              57,49,41,33,25,17,9,1,
              59,51,43,35,27,19,11,3,
              61,53,45,37,29,21,13,5,
              63,55,47,39,31,23,15,7
              };

void converttobinary(int hex[16],int bin[64])
{
int i;

for(i=0;i<64;i++)
bin[i]=0;

long int rem,count;
for(i=0;i<16;i++){
count=1;
 while (hex[i]>0)
{
rem=hex[i]%2;
bin[(i+1)*4-count]=rem;

hex[i]=hex[i]/2;
count++;
}
}
}

void etable()
{
 int i,j,k=0;
 for(i=0;i<8;i++)
 {
   for(j=0;j<6;j++)
   {
```

```
     er[k++]=r[e_bit[i][j]-1];
   }
 }
}




void xor48()
{
  int i;
  for(i=0;i<48;i++)
    {
      if(er[i]==pc2_key[i])
      er[i]=0;
      else
      er[i]=1;
    }
}

int main()
{
  int i,j,k=0,n;
  int hex[64],bin[64],bin1[64];
  FILE *input,*myfile1;
  FILE *key;
  input=fopen("plain.txt","r");


 for(i=0;i<16;i++)
 fscanf(input,"%x",&hex[i]);

converttobinary(hex,bin);
printf("Input from i-1th round\n");
for(i=0;i<64;i++)
    printf("%d",bin[i]);
printf("\n");
printf("After initial permutation of input\n");
for(i=0;i<8;i++)
for(j=0;j<8;j++)
bin1[k++]=bin[ip[i][j]];
for(i=0;i<64;i++)
printf("%d",bin1[i]);
j=0;
printf("\n");

//for next lab program
/* FILE *fp;
for(i=0;i<32;i++)
l[j++]=bin1[i];
fp=fopen("left.txt","w"); */


for(i=0;i<32;i++)
fprintf(fp,"%d",l[i]);
```

```
printf("Right most 32 bits of input plain text\n");
j=0;
for(i=32;i<64;i++)
r[j++]=bin1[i];
for(i=0;i<32;i++)
    printf("%d",r[i]);

 etable();
 printf("\n");
 printf("input after expansion table\n");
 for(i=0;i<48;i++)
    printf("%d",er[i]);

 key=fopen("input.txt","r");
 printf("\ntaking the sub key -1 \n");

    printf("\n");
   for(j=0;j<48;j++)
   fscanf(key,"%1d",&pc2_key[j]);

 for(j=0;j<48;j++)
 printf("%d",pc2_key[j]);


 printf("\n");
 myfile1=fopen("output.txt","w");
xor48();
printf("After xor\n");
for(i=0;i<48;i++)
{
    printf("%d",er[i]);
    fprintf(myfile1,"%d",er[i]);
}
fclose(key);
fclose(input);
 return 0;
}
```

LAB-7

```
#include<stdio.h>
int sbox[32],sbp[32],s[8][6],si=0,l[32],r[32],er[48];


int sbox_table[8][4][16]={
14,4,13,1,2,15,11,8,3,10,6,12,5,9,0,7,
0,15,7,4,14,2,13,1,10,6,12,11,9,5,3,8,
4,1,14,8,13,6,2,11,15,12,9,7,3,10,5,0,
15,12,8,2,4,9,1,7,5,11,3,14,10,0,6,13,
15,1,8,14,6,11,3,4,9,7,2,13,12,0,5,10,
3,13,4,7,15,2,8,14,12,0,1,10,6,9,11,5,
0,14,7,11,10,4,13,1,5,8,12,6,9,3,2,15,
13,8,10,1,3,15,4,2,11,6,7,12,0,5,14,9,
10,0,9,14,6,3,15,5,1,13,12,7,11,4,2,8,
```

```
13,7,0,9,3,4,6,10,2,8,5,14,12,11,15,1,
13,6,4,9,8,15,3,0,11,1,2,12,5,10,14,7,
1,10,13,0,6,9,8,7,4,15,14,3,11,5,2,12,
7,13,14,3,0,6,9,10,1,2,8,5,11,12,4,15,
13,8,11,5,6,15,0,3,4,7,2,12,1,10,14,9,
10,6,9,0,12,11,7,13,15,1,3,14,5,2,8,4,
3,15,0,6,10,1,13,8,9,4,5,11,12,7,2,14,
2,12,4,1,7,10,11,6,8,5,3,15,13,0,14,9,
14,11,2,12,4,7,13,1,5,0,15,10,3,9,8,6,
4,2,1,11,10,13,7,8,15,9,12,5,6,3,0,14,
11,8,12,7,1,14,2,13,6,15,0,9,10,4,5,3,
12,1,10,15,9,2,6,8,0,13,3,4,14,7,5,11,
10,15,4,2,7,12,9,5,6,1,12,14,0,11,3,8,
9,14,15,5,2,8,12,3,7,0,4,10,1,12,11,6,
4,3,2,12,9,5,15,10,11,14,1,7,6,0,8,13,
4,11,2,14,15,0,8,13,3,12,9,7,5,10,6,1,
13,0,11,7,4,9,1,10,14,3,4,12,2,15,8,6,
1,4,11,13,12,3,7,14,10,15,6,8,0,5,9,2,
6,11,13,8,1,4,10,7,9,5,0,15,14,2,3,12,
13,2,8,4,6,15,11,1,10,9,3,14,5,0,12,7,
1,15,13,8,10,3,7,4,12,5,6,11,0,14,9,2,
7,11,4,1,9,12,14,2,0,6,10,13,15,3,5,8,
2,1,14,7,4,10,8,13,15,12,9,0,3,5,6,11
};


int ptable[8][4]={

16,7,20,21,29,12,28,17,
1,15,23,26,5,18,31,10,
2,8,24,14,32,27,3,9,
19,13,30,6,22,11,4,25
};


void sbpermute()
{
int i,j,k=0;
for(i=0;i<8;i++)
{
for(j=0;j<4;j++)
{
sbp[k++]=sbox[ptable[i][j]-1];
}
}
}
void xor32()
{
int i;
for(i=0;i<32;i++)
{
if(l[i]==sbp[i])
r[i]=0;
else
r[i]=1;
}
}
```

```
void dec_bin(int n)
{
int a[4],i=3,j,rem=0;
for(j=0;j<4;j++)
a[j]=0;
while(n!=0)
{
a[i]=n%2;
i--;
n/=2;
}

for(j=0;j<4;j++)
{
sbox[si++]=a[j];
}

}


void sboxf()
{
    printf("\nAfter grouping 6 bits for S box\n");
int i,j,k=0,row,col;
for(i=0;i<8;i++)
{
for(j=0;j<6;j++)
{
    s[i][j]=er[k++];
}
}


for(i=0;i<8;i++)
{
for(j=0;j<6;j++)
{
   printf("%d",s[i][j]);
}
printf("\t");
}

k=0;

for(i=0;i<8;i++)
{
row=(s[i][0]<<1)+s[i][5];
col=(s[i][1]<<3)+(s[i][2]<<2)+(s[i][3]<<1)+s[i][4];
dec_bin(sbox_table[k++][row][col]);
}

}
```

```
void main()
{
int i,j;
FILE *fp;
fp=fopen("input.txt","r");
for(i=0;i<48;i++)
fscanf(fp,"%1d",&er[i]);
printf("\ninput for sbox\n");
for(i=0;i<48;i++)
printf("%d",er[i]);

sboxf();

printf("\nafter sbox\n");

for(i=1;i<=32;i++)
{
printf("%d",sbox[i-1]);
if(i%4==0)
printf(" ");
}

sbpermute();

printf("\n after permutation\n");
for(i=1;i<=32;i++)
{
printf("%d",sbp[i-1]);

}
fp=fopen("left.txt","r");
for(i=0;i<32;i++)
fscanf(fp,"%1d",&l[i]);
printf("\nleft 32 bits of input\n");
for(i=0;i<32;i++)
printf("%d",l[i]);

xor32();
printf("\n after xor-32\n");
for(i=1;i<=32;i++)
{
printf("%d",r[i-1]);

}
}
```