/\*

Anudish Jinturkar

H-27

Subject: Network Security (Lab)

Lab3: Implementation of PlayFair Cipher for character input

\*/

#include <bits/stdc++.h>

#define gc getchar\_unlocked

#define SIZE 5

using namespace std;

const char\* abcstring="ABCDEFGHIJKLMNOPQRSTUVWXYZ";

const char\* extrachar="X";

char mat[SIZE][SIZE];

int a[26];

//to get rid of unwanted chars while decryption

vector<int> unwanted\_char;

int exist(char ch)

{

if(a[(int)ch-65]&&ch>=65&&ch<=97)

{

return 1;

}

else

return 0;

}

int next\_free\_ele()

{

static int i=0,j=0;

for (i=0; i < 26; ++i)

{

if(a[i]==0)

{

a[i]=1;

return abcstring[i];

}

}

}

void visit\_ele(char ch)

{

if(ch>='A'&&ch<='Z')

{

a[(int)ch-65]=1;

}

}

string modify\_string(string s)

{

for(int i=0,j=1;j<s.length();i+=2,j+=2)

{

if(s[i]!=s[j])

continue;

else

{

unwanted\_char.push\_back(j);

s.insert(j,extrachar);

}

}

if(s.length()%2!=0)

{

s+='X';

unwanted\_char.push\_back(s.length()-1);

}

return s;

}

string get\_encrypted\_char(char x,char y,int inc)

{

int row1,col1,row2,col2;

string cycle\_str="";

int ele\_found=0;

for(int i=0;i<SIZE&&ele\_found<2;i++)

{

for(int j=0;j<SIZE&&ele\_found<2;j++)

{

if(mat[i][j]==x)

{

row1=i;

col1=j;

ele\_found++;

}

else if(mat[i][j]==y)

{

row2=i;

col2=j;

ele\_found++;

}

}

}

//take next character in right to left fashion

if(row1==row2)

{

if(col1==SIZE-1&&inc==1)

col1=0;

else if(inc==1)

col1++;

else if(col1==0&&inc==-1)

col1=SIZE-1;

else if(inc==-1)

col1--;

cycle\_str+=mat[row1][col1];

}

//take the next character in up to down fashion

else if(col1==col2)

{

if(row1==SIZE-1&&inc==1)

row1=0;

else if(inc==1)

row1++;

if(row1==0&&inc==-1)

row1=SIZE-1;

else if(inc==-1)

row1--;

cycle\_str+=mat[row1][col1];

}

else

cycle\_str=mat[row1][col2];

return cycle\_str;

}

string remove\_extra\_char(string str)

{

string ret="";

for(int i=0,j=0;i<str.length();i++)

{

if(unwanted\_char[j]==i)

{

j++;

continue;

}

ret+=str[i];

}

return ret;

}

string res;

//s1 modified string

string s,s1;

void play\_fair(int en\_de)

{

res="";

for (int i = 0; i < s1.length(); i+=2)

{

res+=get\_encrypted\_char(s1[i],s1[i+1],en\_de);

res+=get\_encrypted\_char(s1[i+1],s1[i],en\_de);

}

}

int main(){

string enword;

int k=0;

cout<<"Enter the string to encrypt"<<endl;

getline(cin,enword);

cout<<"\n"<<"Enter the keyword(only characters)"<<endl;

getline(cin,s);

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

{

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

{

if(k<s.length())

{

if(exist(s[k]))

{

k++;

continue;

}

mat[i][j]=s[k];

visit\_ele(s[k]);

k++;

}

else

{

mat[i][j]=next\_free\_ele();

}

}

}

s1=modify\_string(enword);

play\_fair(1);

cout<<"\n"<<"Encrypted string is "<<endl;

cout<<res<<endl;

s1=res;

play\_fair(-1);

res=remove\_extra\_char(res);

cout<<"\n"<<"Decrypted string is"<<endl;

cout<<res<<endl;

return 0;

}

/\*

OUTPUT :

Enter the string to encrypt

ANUDISH

Enter the keyword(only characters)

JINT

Encrypted string is

JTWBAPLV

Decrypted string is

ANUDISH

\*/