ADFGVX密码算法

李奕萱 151160030

古典密码：德国陆军强于恺撒

◆ The German ADFGVX Field Cipher

	A	D	F	G	V	X
A	*B*	3	*M*	*R*	*L*	*I*
D	*A*	6	*F*	ϕ	8	2
F	*C*	7	*S*	*E*	*U*	*H*
G	*Z*	9	*D*	*X*	*K*	*V*
V	1	*Q*	*Y*	*W*	5	*P*
X	*N*	*J*	*T*	4	*G*	*O*

你试试：

XFVAXAFFGXFFGXFXFXXGDXDAFGVA

Plaintext: **field**

cipher

DF AX FG AV GF FA AX VX FX FG AG

R	I	F	L	E
D	F	A	X	X
A	G	F	X	F
F	X	F	G	V
G	F	G	D	A
V	X	X	X	X
A	F	F		

```
string input;//被加密的数据
string pkey;//移位密码
string sub_encryption;//中间加密码
string encryption;//最终加密码
string sort_pkey;//按字典序排序的移位密码
bool flag;
bool f[100];
char secret_table[36]={//分解密码表
    'B','3','M','R','L','I',
    'A','6','F','$','8','2',
    'C','7','S','E','U','H',
    'Z','9','D','X','K','V',
    '1','Q','Y','W','5','P',
    'N','J','T','4','J','O',
};
char ADFGVX[6]={'A','D','F','G','V','X'};//
int main(){
    cout<<"Please input the original sentence and the pkey."<<endl;
    cin>>input>>pkey;
    int len_origin=(int)input.size();//被加密码长度
int len_pkey=(int)pkey.size();//移位密码长度
int len_encryption=2*len_origin;//加密码长度
sort_pkey=pkey;//对移位密码进行排序
sort(sort_pkey.begin(),sort_pkey.end());
int i,j,k,m,num;
for(i=0;i<len_origin;i++){//对原字符串进行转换
    flag=false;
    for(j=0;j<36;j++){//在分解密码表中搜索
        if(input[i]==secret_table[j]){
            flag=true;
            sub_encryption[2*i]=ADFGVX[j/6];
            sub_encryption[2*i+1]=ADFGVX[j%6];
        }
    }
    if(flag==false){
        cout<<"Bad expression"<<endl;
        return 0;
    }
}
```

```
int nb_line=len_encryption/len_pkey;//在移位密码表中对应的行数
int r=len_encryption%len_pkey;//在移位密码表中对应的余数
memset(f,false,sizeof(f));
for(i=0;i<len_pkey;i++){//用于计算移位密码中对应的列的元素数目。
    for(j=0;j<len_pkey;j++){
        if(sort_pkey[i]==pkey[j]){
            if(j<r){
                f[i]=true;
            }
            break;
        }
    }
}
for(k=0;k<len_encryption;k++){//用于计算中间加密码在最终加密码中的位置
    i=k/len_pkey;//hang
    j=k%len_pkey;//lie
    num=0;
    for(m=0;m<len_pkey;m++){
        if(pkey[j]==sort_pkey[m])break;
    }
    //cout<<m;
    while(m>0){
        m--;
        if(f[m]){
            num=num+nb_line+1;
        }
        else{
            num=num+nb_line;
        }
    }
    num=num+i;
    encryption[num]=sub_encryption[k];
    //cout<<encryption[num]<<" "<<num<<endl;
}
for(i=0;i<len_encryption;i++){
    cout<<encryption[i]<<endl;
}
```

Thank you!