

Assignment 3

1. Playfair Cipher

In this scheme, pairs of letters are encrypted, instead of single letters as in the case of simple substitution cipher.

Initially, a key table is created. The key table is a 5x5 grid of alphabets that acts as the key for encrypting the plaintext. Each of the 25 alphabets must be unique and one letter of the alphabet(usually J) is omitted from the table, as we need only 25 alphabets instead of 26. If the text contains J, then it is replaced by I.

Process of playfair cipher

- A plaintext is split into pairs of two letters (digraphs). If there is an odd number of letters, a Z letter is added to the last letter.
- The rules of encryption are
 - If both the letters are in the same column, take the letter below each one.
 - If both are in same row, then taking right of each one.
 - Else, form a rectangle with the two letters and take the horizontal opposite corner of the rectangle.
- The decryption can also take place by doing these rules in reverse.

Code :

```
#include<bits/stdc++.h>
using namespace std;

pair<int,int> findInTable(char table[5][5], char x)
{
    for(int i=0;i<5;i++)
    {
        for(int j=0;j<5;j++)
        {
            if(table[i][j]==x)
            {
                pair<int,int>ans = make_pair(i,j);
                return ans;
            }
        }
    }
}

int main()
{
    string key = "monarchy";
    int i,j;
    /// Generating table
    char table[5][5];

    bool arr[26];
    memset(arr,false,sizeof(arr));
    int index = 0;
    for(i=0;i<key.length();i++)
    {
        if(arr[key[i]-'a']==false && key[i]!='j')
        {
            table[index/5][index%5] = key[i];
            arr[key[i]-'a'] = true;
        }
    }
}
```

```

        index++;
    }
}
for(i=0;i<26;i++)
{
    if('a'+i=='j') continue;
    if(arr[i]==false)
    {
        table[index/5][index%5] = 'a'+i;
        arr[i] = true;
        index++;
    }
}
cout<<"5x5 Square Table : \n";
for(i=0;i<5;i++)
{
    for(j=0;j<5;j++)    cout<<table[i][j]<<" ";
    cout<<"\n";
}
cout<<endl;
string originalText;
cout<<"Enter PlainText : ";
cin>>originalText;

int len = originalText.length();

/// Converting to lowercase(if in uppercase)
for(i=0;i<len;i++)
{
    if(originalText[i]>='A' && originalText[i]<='Z')
    {
        originalText[i] = originalText[i]-'A'+ 'a';
    }
}

if(len%2==1)
{
    originalText += 'z';
    len++;
}
string EncryptedText="";
for(i=0;i<len;i+=2)
{
    char a1 = originalText[i];
    char a2 = originalText[i+1];

    if(a1=='j')        a1='i';
    else if(a2=='j')    a2 = 'i';

    pair<int,int>p1 = findInTable(table, a1);
    pair<int,int>p2 = findInTable(table, a2);

    if(p1.second==p2.second)
    {
        EncryptedText += table[(p1.first+1)%5][p1.second];
        EncryptedText += table[(p2.first+1)%5][p2.second];
    }
    else if(p1.first==p2.first)
    {
        EncryptedText += table[p1.first][(p1.second+1)%5];
        EncryptedText += table[p1.first][(p2.second+1)%5];
    }
    else
    {
        EncryptedText += table[p1.first][p2.second];
    }
}

```

```

        EncryptedText += table[p2.first][p1.second];
    }
}

cout<<"The Encrypted String is : "<<EncryptedText<<endl;

string DecryptedText = "";

for(i=0;i<len;i+=2)
{
    char a1 = EncryptedText[i];
    char a2 = EncryptedText[i+1];

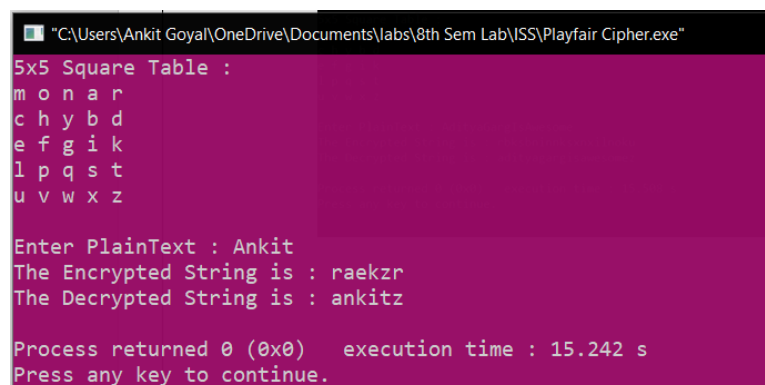
    pair<int,int>p1 = findInTable(table,a1);
    pair<int,int>p2 = findInTable(table,a2);

    if(p1.second==p2.second)
    {
        DecryptedText += table[(p1.first-1 + 5)%5][p1.second];
        DecryptedText += table[(p2.first-1 + 5)%5][p2.second];
    }
    else if(p1.first==p2.first)
    {
        DecryptedText += table[p1.first][(p1.second-1+5)%5];
        DecryptedText += table[p2.first][(p2.second-1+5)%5];
    }
    else
    {
        DecryptedText += table[p1.first][p2.second];
        DecryptedText += table[p2.first][p1.second];
    }
}

cout<<"The Decrypted String is : "<<DecryptedText<<endl;
}

```

Result :



```

"C:\Users\Ankit Goyal\OneDrive\Documents\labs\8th Sem Lab\ISS\Playfair Cipher.exe"
5x5 Square Table :
m o n a r
c h y b d
e f g i k
l p q s t
u v w x z

Enter PlainText : Ankit
The Encrypted String is : raekzr
The Decrypted String is : ankitz

Process returned 0 (0x0)   execution time : 15.242 s
Press any key to continue.

```

2. Hill Cipher

This is a polygraphic substitution cipher based on linear algebra. Each letter is represented by a number modulo 26. The characters are mapped by the scheme as A with 0, B with 1, ... , Z with 25.

To encrypt a message, each block of n letters is multiplied by an invertible $n \times n$ matrix, against modulo 26. To decrypt a message, each block is multiplied by the inverse of the matrix used for encryption. The matrix used for encryption is the cipher key, and it should be chosen randomly from the set of invertible $n \times n$ matrix (modulo 26).

Here is the code for $n = 3$.

Hence, there will be the plaintext of only 3 length.

Code :

```
#include <bits/stdc++.h>
using namespace std;

string lowercase(string s)
{
    for(int i =0;i<s.length();i++)
    {
        if(s[i]>='A' && s[i]<='Z')
        {
            s[i] = s[i]-'A'+'a';
        }
    }
    return s;
}

int main()
{
    string key = "GYBNQKURP";
    key = lowercase(key);

    int table[3][3],i,j;
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
            table[i][j] = key[i*3+j]-'a';

    cout<<"Encryption Table : \n";
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
            cout<<table[i][j]<<" ";
        cout<<"\n";
    }
    cout<<endl;
    cout<<"Decryption Table : \n";
    /// Decryption
    int DecryptTable[3][3];
    for(i = 0; i < 3; i++){
        for(j = 0; j < 3; j++)
        {
            DecryptTable[i][j] = (-
1*((table[(j+1)%3][(i+1)%3]*table[(j+2)%3][(i+2)%3]) - (table[(j+1)%3][(i+2)%3] *
table[(j+2)%3][(i+1)%3]))%26+26)%26;
            cout<<DecryptTable[i][j]<<" ";
        }
        cout<<"\n";
    }
```

```

    }
    cout<<endl;
    string OriginalString;
    cout<<"Enter string of length 3 : ";
    cin>>OriginalString;
    OriginalString = lowercase(OriginalString);

    int OriginalArray[3];
    for(i=0;i<3;i++)
    {
        OriginalArray[i] = OriginalString[i] - 'a';
    }

    int EncryptedArray[3];
    string EncryptedText = "";
    for(i=0;i<3;i++)
    {
        EncryptedArray[i] = (OriginalArray[0]*table[i][0] +
        OriginalArray[1]*table[i][1] + OriginalArray[2]*table[i][2])%26;
        EncryptedText += 'a'+EncryptedArray[i];
    }
    cout<<"Encryped Text : "<<EncryptedText;
    cout<<endl;

    string DecryptedString = "";
    for(i=0;i<3;i++)
    {
        DecryptedString += 'a' + (EncryptedArray[0]*DecryptTable[i][0] +
        EncryptedArray[1]*DecryptTable[i][1] + EncryptedArray[2]*DecryptTable[i][2])%26;
    }

    cout<<"Decrypted String : "<<DecryptedString;
    cout<<endl;
}

```

Result :

```

"C:\Users\Ankit Goyal\OneDrive\Documents\labs\8th Sem Lab\ISS\Hill Cipher.exe"
Encryption Table :
6 24 1
13 16 10
20 17 15

Enter string of length 3 : ajf
Decryption Table :
8 5 10
21 8 21
21 12 8

Encryped Text : nmv
Decrypted String : ajf

Process returned 0 (0x0)   execution time : 2.938 s
Press any key to continue.

```