**Experiment-3**

Aim: Write a program to implement playfair cipher.

Source Code:

#include<bits/stdc++.h>

using namespace std;

#define SIZE 30

char ps, ks, keyT[5][5];

void toLowerCase(char plain[], int ps)

{

int i;

for (i = 0; i < ps; i++) {

if (plain[i] > 64 && plain[i] < 91)

plain[i] += 32;

}

}

int removeSpaces(char\* plain, int ps)

{

int i, count = 0;

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

if (plain[i] != ' ')

plain[count++] = plain[i];

plain[count] = '\0';

return count;

}

void generateKeyTable(char key[], int ks, char keyT[5][5])

{

int i, j, k, \*dicty;

dicty = (int\*)calloc(26, sizeof(int));

for (i = 0; i < ks; i++) {

if (key[i] != 'j')

dicty[key[i] - 97] = 2;

}

dicty['j' - 97] = 1;

i = 0;

j = 0;

for (k = 0; k < ks; k++) {

if (dicty[key[k] - 97] == 2) {

dicty[key[k] - 97] -= 1;

keyT[i][j] = key[k];

j++;

if (j == 5) {

i++;

j = 0;

}

}

}

for (k = 0; k < 26; k++) {

if (dicty[k] == 0) {

keyT[i][j] = (char)(k + 97);

j++;

if (j == 5) {

i++;

j = 0;

}

}

}

}

void search(char keyT[5][5], char a, char b, int arr[])

{

int i, j;

if (a == 'j')

a = 'i';

else if (b == 'j')

b = 'i';

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

for (j = 0; j < 5; j++) {

if (keyT[i][j] == a) {

arr[0] = i;

arr[1] = j;

}

else if (keyT[i][j] == b) {

arr[2] = i;

arr[3] = j;

}

}

}

}

int mod5(int a)

{

return ((a%5+5)%5);

}

int prepare(char str[], int ptrs)

{

if (ptrs % 2 != 0) {

str[ptrs++] = 'x';

str[ptrs] = '\0';

}

return ptrs;

}

void encrypt(char str[], char keyT[5][5], int ps)

{

int i, a[4];

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

search(keyT, str[i], str[i + 1], a);

if (a[0] == a[2]) {

str[i] = keyT[a[0]][mod5(a[1] + 1)];

str[i + 1] = keyT[a[0]][mod5(a[3] + 1)];

}

else if (a[1] == a[3]) {

str[i] = keyT[mod5(a[0] + 1)][a[1]];

str[i + 1] = keyT[mod5(a[2] + 1)][a[1]];

}

else {

str[i] = keyT[a[0]][a[3]];

str[i + 1] = keyT[a[2]][a[1]];

}

}

}

void decrypt(char str[])

{

int i, a[4];

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

search(keyT, str[i], str[i + 1], a);

if (a[0] == a[2]) {

str[i] = keyT[a[0]][mod5(a[1] - 1)];

str[i + 1] = keyT[a[0]][mod5(a[3] - 1)];

}

else if (a[1] == a[3]) {

str[i] = keyT[mod5(a[0] - 1)][a[1]];

str[i + 1] = keyT[mod5(a[2] - 1)][a[1]];

}

else {

str[i] = keyT[a[0]][a[3]];

str[i + 1] = keyT[a[2]][a[1]];

}

}

}

void encryptByPlayfairCipher(char str[], char key[])

{

ks = strlen(key);

ks = removeSpaces(key, ks);

toLowerCase(key, ks);

ps = strlen(str);

toLowerCase(str, ps);

ps = removeSpaces(str, ps);

ps = prepare(str, ps);

generateKeyTable(key, ks, keyT);

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

{

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

cout<<keyT[i][j]<<" ";

cout<<endl;

}

encrypt(str, keyT, ps);

}

int main()

{

char str[SIZE], key[SIZE];

cout<<"Enter key text: ";

cin.getline(key,SIZE);

cout<<"Enter plain text: ";

cin.getline(str,SIZE);

encryptByPlayfairCipher(str, key);

printf("Cipher text: %s\n", str);

cout<<"Decrypted Cipher text: "<<endl;

decrypt(str);

cout<<str;

return 0;

}

**Output:**

Enter key text: Monarchy

Enter plain text: gatlmzclrqtx

Cipher text: inskrumentsz

Decrypted Cipher text: gatlmzclrqtx

**Experiment 4**

Aim: Write a program to encrypt and decrypt a message using hill cipher.

Source Code:

**Encryption**

#include <iostream>

#include<string>

#include <cstdlib>

using namespace std;

void func(int x[2][2], string t)

{

int c1[2][1];

int b1[2][1];

char ch1[2][1];

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

{for(int j=0;j<1;j++)

{

b1[i][j]=t[i]-97;

}

}

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

{

for (int j = 0; j < 1; j++)

{

c1[i][j] = 0;

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

{

c1[i][j] = c1[i][j] + x[i][k] \* b1[k][j];

}

}

}

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

{for(int j=0;j<1;j++)

{c1[i][j]=c1[i][j]%26;}

}

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

{for(int j=0;j<1;j++)

{ ch1[i][j]=c1[i][j]+97;

cout<<ch1[i][j];}

cout<<endl;

}

}

int main(int arg, char\* args[]) {

char text1[10];

string text="";

strcpy(text1,args[1]);

text+=text1;

int b1[2][1];

int b2[2][1];

int k=0;

if(strlen(text1)%2!=0)

{text+="x";

strcat(text1,"x");

}

cout<<"plain text: "<<text<<endl;

int a[2][2]={5,3,3,2};

cout<<endl;

cout<<"key"<<endl;

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

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

{cout<<a[i][j];}

cout<<endl;

}

int len=strlen(text1);

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

while(k<len)

{

string b;

b = text.substr(k,2);

func(a,b);

k+=2;

}

}

**Decryption**

#include <iostream>

#include<string>

#include <cstdlib>

#include<cmath>

using namespace std;

void func(int x[2][2], string t)

{

int c1[2][1];

int b1[2][1];

char ch1[2][1];

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

{for(int j=0;j<1;j++)

{

b1[i][j]=t[i]-97;

}

}

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

{

for (int j = 0; j < 1; j++)

{

c1[i][j] = 0;

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

{

c1[i][j] = c1[i][j] + x[i][k] \* b1[k][j];

}

}

}

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

{for(int j=0;j<1;j++)

{

if(c1[i][j]<0)

{c1[i][j]=26+c1[i][j];

if(c1[i][j]<97)

{c1[i][j]=26+c1[i][j];}

}

c1[i][j]=(c1[i][j]%26);}

}

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

{for(int j=0;j<1;j++)

{ ch1[i][j]=c1[i][j]+97;

if(ch1[i][j]<97)

{ch1[i][j]=26+ch1[i][j];}

cout<<ch1[i][j];}

cout<<endl;

}

}

int main(int arg, char\* args[]) {

char text1[10];

string text="";

strcpy(text1,args[1]);

text+=text1;

int b1[2][1];

int b2[2][1];

int k=0;

if(strlen(text1)%2!=0)

{text+="x";

strcat(text1,"x");

}

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

int a[2][2]={5,3,3,2};

cout<<endl;

cout<<"key"<<endl;

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

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

{cout<<a[i][j];}

cout<<endl;

}

int len=strlen(text1);

int ad[2][2]={a[1][1],-a[0][1],-a[1][0],a[0][0]};

int in[2][2];

int det;

det=((a[0][0]\*a[1][1]) - (a[1][0]\*a[0][1]));

det=det%26;

for (int x=1; x<26; x++)

{if ((det\*x) % 26 == 1)

ndet=x;

}

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

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

{in[i][j]=ndet\*ad[i][j];}

}

cout<<"plain text: "<<endl;

while(k<len)

{

string b;

b = text.substr(k,2);

func(in,b);

k+=2;

}

}

**Encryption**

**Decryption**

**Experiment-6**

Aim: Write a program to create a password and determine whether the password is strong or not.

Source Code:

#include <iostream>

using namespace std;

void printStrongLevel(string& input)

{

int n = input.length();

bool hasLower = false, hasUpper = false;

bool hasDigit = false, specialChar = false;

string normalChars = "abcdefghijklmnopqrstu"

"vwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890 ";

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

if (islower(input[i]))

hasLower = true;

if (isupper(input[i]))

hasUpper = true;

if (isdigit(input[i]))

hasDigit = true;

}

unsigned long special = input.find\_first\_not\_of(normalChars);

if (special != string::npos)

specialChar = true;

cout << "Strength of password : ";

if (hasLower && hasUpper && hasDigit &&

specialChar && (n >= 8))

cout << "Strong" << endl;

else if ((hasLower || hasUpper) &&

specialChar && (n >= 6))

cout << "Moderate" << endl;

else

cout << "Weak" << endl;

}

int main()

{

string input;

cout<<"Enter password : ";

cin>>input;

printStrongLevel(input);

return 0;

}

Output:

Enter password : CNS@ssd123

Strength of password : Strong

**Experiment-7**

Aim: To implement rail fence encryption.

Source Code:

#include <iostream>

using namespace std;

int main() {

string s;

cout<<"Enter Plaintext : ";

cin>>s;

int col = s.length()/2;

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

col++;

}

cout<<"Fence : "<<endl;

char fenceMatrix[2][col];

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

fenceMatrix[0][i/2] = s[i];

fenceMatrix[1][i/2] = s[i+1];

}

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

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

cout<<fenceMatrix[i][j]<<" ";

}

cout<<endl;

}

cout<<"Encrypted Message : ";

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

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

cout<<fenceMatrix[i][j];

}

}

return 0;

}

Output :

Enter Plaintext : cryptography

Fence :

c y t g a h

r p o r p y

Encrypted Message : cytgahrporpy