**Submitted by – Paras Jain (2018kucp1006) ISS Lab-2**

**Q1) a) Code:**

#include<bits/stdc++.h>

using namespace std;

int a = 15, b, m= 26,x;

int main(){

    cout<<"Enter value of b\n";

    cin>>b;

    b=b%m;

    int inver\_a = 0;

    for(int i=0;i<m;i++){

        if((i\*a)%26 == 1){

            inver\_a = i;  // finding inverse of a by using formula (inv(a)\*a)%m =1

            break;

        }

    }

    while(1){

        string plain\_text, cipher;

        cout << "To Encrypt text press 1\nTo Decrypt text press 2\nTo exit press 0\n";

        cin >> x;

// Encryption code

        if(x==1){

            cout << "Enter plain text\n";

            cin >> plain\_text;

            for (int i = 0; i < plain\_text.length(); i++){

                int temp = plain\_text[i]-'a'; // converting to integral value

                temp = (temp\*a + b) %m; //applying formula

                char ch = 'a' + temp; //converting to ascii value

                cipher.push\_back(ch);

                  // Generating cipher text by pushing characters

            }

            cout << "Encrypted Message is\n"

                 << cipher << endl;

        }

// Decription Code

        else if(x==2){

            cout << "Enter cipher text\n";

            cin >> cipher;

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

                int temp = cipher[i]-'a'; // converting to integral value

                temp = ((temp - b + m)\*inver\_a) %m; //applying formula

                char ch = 'a' + temp; //converting to ascii value

                plain\_text.push\_back(ch);

                  // Generating cipher text by pushing characters

            }

            cout << "Decrypted Message is\n"

                 << plain\_text << endl;

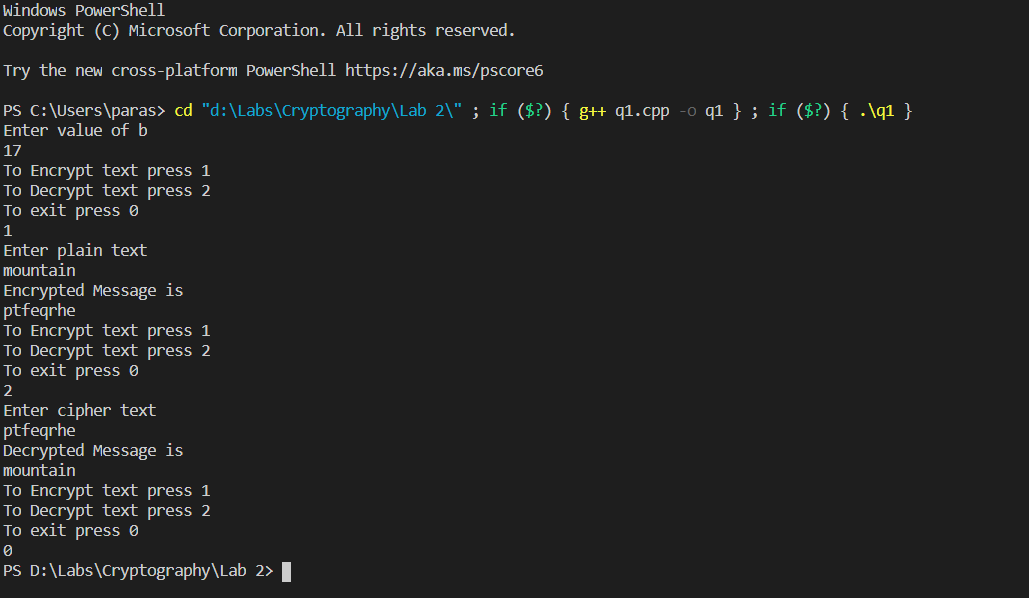
        }

        else break;

    }

}

**Output:**



**b)**

No, B can take any value. All the values of B modulo 26 are equivalent. So if B is negative, there is an equivalent positive value of B. Example: 'B = -1' is equivalent to 'B = 25' (modulo 26)

**c)**

‘a’ should be chosen to be relatively prime to m because a\*p could be the multiple of 26 and hence modulo will be ‘b’ for multiple possible case and hence it would fail.

**Q2.**

**Code:**

//Q2 Playfair cipher

//Submitted by Paras Jain

#include <bits/stdc++.h>

using namespace std;

char mat[5][5];

//setting key

void generateMatrix(string key){

    map<char,bool> mp;

    string new\_key;     //this will contain 26 characters first the unique key characters and then remaining alphabets

    // filling key characters uniquely in new\_key

    for(int i=0;i<key.length();i++){

        if(mp.find(key[i])==mp.end()){

            if(key[i]=='i' or key[i]=='j'){

                new\_key.push\_back('i');

                mp['i']=true;           //i and j are single unit so if anyone comes mark them true

                mp['j']=true;

            }

            else{

                new\_key.push\_back(key[i]);

                mp[key[i]] = true;              //visited character is marked as true.

            }

        }

    }

    // filling the remaining characters in new\_key

    for(char a='a';a<='z';a++){

        if (mp.find(a) == mp.end())

        {

            if (a == 'i' or a == 'j')

            {

                new\_key.push\_back('i');

                mp['i'] = true;

                mp['j'] = true;

            }

            else

            {

                new\_key.push\_back(a);

                mp[a] = true;

            }

        }

    }

    int pos=0;

    cout<<"Matrix is:\n";

    // filling the characters in matrix

    for(int i=0;i<5;i++){

        for(int j=0;j<5;j++){

            mat[i][j] = new\_key[pos];

            cout<<mat[i][j]<<" ";

            pos++;

        }

        cout<<endl;

    }

}

string encrypt(char a,char b){

    string temp;

    int ia,ib,ja,jb;

    //finding the position of a nd b in the matrix

    for(int i=0;i<5;i++){

        for(int j=0;j<5;j++){

            if(mat[i][j]==a){

                ia=i;

                ja=j;

            }

            else if(mat[i][j]==b){

                ib=i;

                jb=j;

            }

        }

    }

    // if they are in same row case

    if(ia==ib){

        temp.push\_back(mat[ia][(ja+1)%5]); //inserting the next character

        temp.push\_back(mat[ib][(jb+1)%5]);

        return temp;

    }

    //if they are in same column case

    else if(ja==jb){

        temp.push\_back(mat[(ia + 1) % 5][ja]); //inserting the next character

        temp.push\_back(mat[(ib + 1) % 5][jb]);

        return temp;

    }

    // neither same row nor same column case

    else{

        temp.push\_back(mat[ia][jb]);

        temp.push\_back(mat[ib][ja]);

        return temp;

    }

}

string decrypt(char a, char b){

    string temp;

    int ia, ib, ja, jb;

    //finding position in the matrix of a and b

    for (int i = 0; i < 5; i++){

        for (int j = 0; j < 5; j++){

            if (mat[i][j] == a){

                ia = i;

                ja = j;

            }

            else if (mat[i][j] == b){

                ib = i;

                jb = j;

            }

        }

    }

    //if same row case

    if (ia == ib){

        temp.push\_back(mat[ia][(ja + 4 ) % 5]); // 4 = ja -1 + 5 as it can be negative

        temp.push\_back(mat[ib][(jb + 4 ) % 5]);

        return temp;

    }

    // if same column case

    else if (ja == jb){

        temp.push\_back(mat[(ia + 4) % 5][ja]); // 4 = ja -1 + 5 as it can be negative

        temp.push\_back(mat[(ib + 4) % 5][jb]);

        return temp;

    }

    //neither same row nor same column case

    else{

        temp.push\_back(mat[ia][jb]);

        temp.push\_back(mat[ib][ja]);

        return temp;

    }

}

int main(){

    int x;

    cout << "Enter key\n";

    string key;

    cin >> key;

    generateMatrix(key);    // function to generate 5X5 matrix to set the key

    while (1){

        string plain\_text, cipher;

        cout << "To Encrypt text press 1\nTo Decrypt text press 2\nTo exit press 0\n";

        cin >> x;       // user choice to encrypt or decrypt

        //Encryption code

        if (x == 1){

            cout << "Enter plain text\n";

            cin >> plain\_text;

            for (int i = 0; i < plain\_text.length(); i++){

                if(plain\_text[i]=='j') plain\_text[i]='i';       // converting j into i as they are used as one unit

                if (i == plain\_text.length() - 1)               // case 1 if last character is single pair it with 'z'

                    cipher+=encrypt(plain\_text[i],'z');

                else if(plain\_text[i]==plain\_text[i+1])         // case to if 2 characters are similar then pair the first one with 'x'

                    cipher+=encrypt(plain\_text[i],'x');

                else{

                    cipher+=encrypt(plain\_text[i],plain\_text[i+1]);       // case 3 pairing the 2 characters

                    i++;

                }

            }

            cout << "Encrypted Message is\n"

                 << cipher << endl;

        }

        //decription part

        else if (x == 2){

            cout << "Enter cipher text\n";

            cin >> cipher;

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

                plain\_text += decrypt(cipher[i], cipher[i + 1]);    // pairing 2 characters and decrypting them

            cout << "Decrypted Message is\n"

                 << plain\_text << endl;

        }

        else

            break;

    }

}

**Output:**

