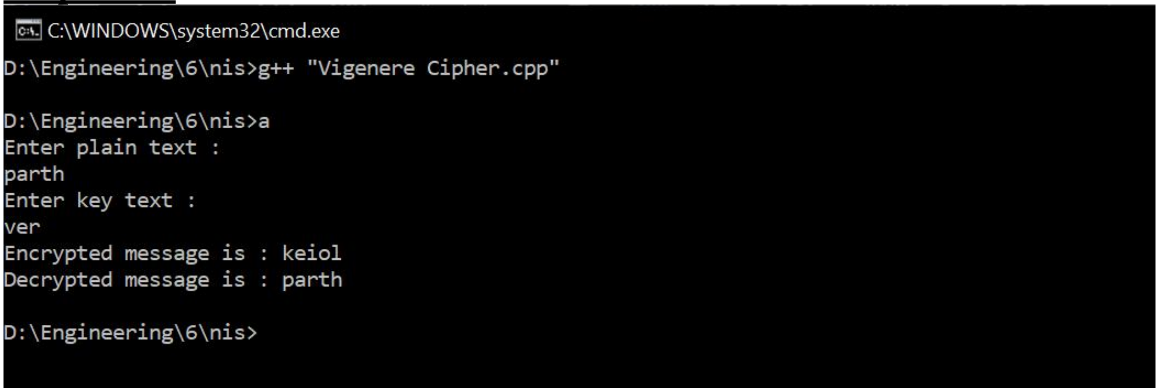
NIS LAB:-3 Assignment

1. Vigenere Cipher :-

```
#include <bits/stdc++.h>
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
#define ll long long
string plain_text, cipher_text, final_text,key;
ll m;
string encrypt(string plain_text, string key,ll m)
{
    cipher_text = plain_text;
    ll i, j;
    for (i = 0; i < plain_text.length(); i++)
    {
        j = ((plain_text[i] - 'a')+(key[i%m] - 'a')) % 26;
        cipher_text[i] = j + 'a';
    }
    return cipher_text;
}
string decrypt(string cipher_text, string key, ll m)
{
    ll i, j;
    final_text = plain_text;
    for (i = 0; i < plain_text.length(); i++)
    {
        j =((cipher_text[i] - 'a')-(key[i%m] - 'a')) % 26 ;
        if (j < 0)
        {
            j += 26;
        }
        final_text[i] = j + 'a';
    }
    return final_text;
}

int main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    cout << "Enter plain text : " << endl;
    getline(cin,plain_text);
    cout << "Enter key text : " << endl;
    cin >> key;
    m = key.length();
    cipher_text = encrypt(plain_text, key,m);
    cout << "Encrypted message is : " << cipher_text << endl;
    final_text = decrypt(cipher_text, key, m);
    cout << "Decrypted message is : " << final_text << endl;
    return 0;
}
```

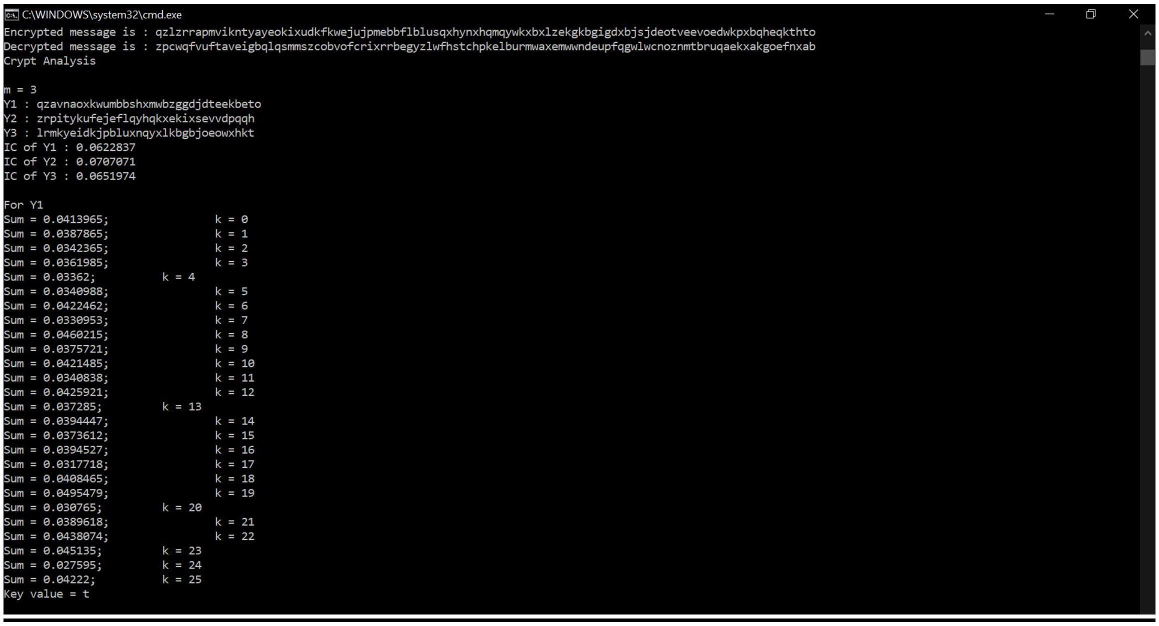
sanpShot:-



```
D:\Engineering\6\nis>g++ "Vigenere Cipher.cpp"

D:\Engineering\6\nis>a
Enter plain text :
parth
Enter key text :
ver
Encrypted message is : keiol
Decrypted message is : parth

D:\Engineering\6\nis>
```

2. Cryptanalysis of Vigenere Cipher:-

```
#include <bits/stdc++.h>
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
#define ll long long
string plain_text, cipher_text, final_text, key;
ll m;
string encrypt(string plain_text, string key, ll m)
{

    cipher_text = plain_text;
    ll i, j;

    for (i = 0; i < plain_text.length(); i++)
    {
        j = ((plain_text[i] - 'a') + (key[i % m] - 'a')) % 26;
        cipher_text[i] = j + 'a';
    }
    return cipher_text;
}
string decrypt(string cipher_text, string key, ll m)
{
    ll i, j;
    final_text = plain_text;
    for (i = 0; i < plain_text.length(); i++)
    {
        j = ((cipher_text[i] - 'a') - (key[i % m] - 'a')) % 26;
        if (j < 0)
        {
            j += 26;
        }
        final_text[i] = j + 'a';
    }
    return final_text;
}
void cryptAnalysis(string ct)
{
    float english_freq[] = {8.167, 1.492, 2.782, 4.253, 12.702, 2.228, 2.015, 6.094, 6.996, 0.153, 0.772, 4.025, 2.406, 6.749, 7.507, 1.929, 0.095, 5.987, 6.327, 9.056, 2.758, 0.978, 2.360, 0.150, 1.974, 0.074};
    float p[26];

    for (int i = 0; i < 26; i++)
    {
        p[i] = english_freq[i] / 100;
    }
    for (int m = 3; m < ct.length() / 3; m++)
    {
        bool isCorrectLen = true;
        vector<char> Y[m];
        for (int i = 0; i < ct.length(); i++)
        {
            Y[i % m].push_back(ct.at(i));
        }
        cout << "\nm = " << m << "\n";
        for (int i = 0; i < m; i++)
        {
            cout << "Y" << i + 1 << " : ";
            for (int j = 0; j < Y[i].size(); j++)
            {
                cout << Y[i][j];
            }
            cout << "\n";
        }
        float icsum[m] = {0.0};
        int freq[26] = {0};
        float q[m][26];
        for (int j = 0; j < m; j++)
        {
            for (int i = 0; i < 26; i++)
            {
                freq[i] = 0;
            }

            for (int i = 0; i < Y[j].size(); i++)
            {
                freq[Y[j][i] - 'a'] += 1;
            }

            for (int i = 0; i < 26; i++)
            {
                q[j][i] = (float)(freq[i]) / Y[j].size();
            }

            for (int i = 0; i < 26; i++)
            {
                icsum[j] += q[j][i] * q[j][i];
            }
            cout << "IC of Y" << j + 1 << " : " << icsum[j] << "\n";

            if (icsum[j] < 0.06)
            {
                isCorrectLen = false;
                break;
            }
        }
        if (isCorrectLen)
        {
            char key[m];
            for (int j = 0; j < m; j++)
            {
                cout << "\nFor Y" << j + 1 << "\n";
                vector<float> sum(26);
                for (int k = 0; k < 26; k++)
                {
                    sum[k] = 0.0f;
                    for (int i = 0; i < 26; i++)
                    {
                        sum[k] += (p[i] * q[j][(i + k) % 26]);
                    }
                    cout << "Sum = " << sum[k] << ";\t\tk = " << k << "\n";
                }
                int index = max_element(sum.begin(), sum.end()) - sum.begin();
                key[j] = (index + 'a');
                cout << "Key value = " << key[j] << "\n";
                cout << "\n";
            }
            cout << "Key = ";
            for (int i = 0; i < m; i++)
            {
                cout << key[i];
            }
            cout << "\n";
            break;
        }
    }
}

int main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);

    cout << "Enter plain text    : " << endl;
    getline(cin, plain_text);

    cout << "Enter key text    : " << endl;
    cin >> key;

    m = key.length();

    cipher_text = encrypt(plain_text, key, m);
    cout << "Encrypted message is : " << cipher_text << endl;

    final_text = decrypt(cipher_text, key, m);
    cout << "Decrypted message is : " << final_text << endl;

    cout << "Crypt Analysis" << endl;
    cryptAnalysis(cipher_text);
    return 0;
}
```



```
Encrypted message is : qzlzrrapmvikntyayeokixudkfkwejujpmebbflblusqxhynxhqmqywkxbxlzekgkbgigdxbjsjdeotveevoedwkpxbqheqkthto
Decrypted message is : zpcwqfvuftaveigbqlqsmmszcobvofcrixrrbegyzlwfhstchpkelburmwaxemwwndeupfqgwlwcnoznmtbruqaekxakgoefnxab
Crypt Analysis

m = 3
Y1 : qzavnaoxkwumbbshxmwbzggdjdteekbeto
Y2 : zrpitykufejeflqyhqkxekixsevvdpqqh
Y3 : lrmkyeidkjpbluxnqyxlkbgbjoeowxhkt
IC of Y1 : 0.0622837
IC of Y2 : 0.0707071
IC of Y3 : 0.0651974

For Y1
Sum = 0.0413965;                k = 0
Sum = 0.0387865;                k = 1
Sum = 0.0342365;                k = 2
Sum = 0.0361985;                k = 3
Sum = 0.03362;          k = 4
Sum = 0.0340988;                k = 5
Sum = 0.0422462;                k = 6
Sum = 0.0330953;                k = 7
Sum = 0.0460215;                k = 8
Sum = 0.0375721;                k = 9
Sum = 0.0421485;                k = 10
Sum = 0.0340838;                k = 11
Sum = 0.0425921;                k = 12
Sum = 0.037285;         k = 13
Sum = 0.0394447;                k = 14
Sum = 0.0373612;                k = 15
Sum = 0.0394527;                k = 16
Sum = 0.0317718;                k = 17
Sum = 0.0408465;                k = 18
Sum = 0.0495479;                k = 19
Sum = 0.030765;         k = 20
Sum = 0.0389618;                k = 21
Sum = 0.0438074;                k = 22
Sum = 0.045135;         k = 23
Sum = 0.027595;         k = 24
Sum = 0.04222;          k = 25
Key value = t
```