**Cryptography and Network Security**

**B.Tech. (CSE) – I [ 2022-23 ]**

**Assignment No - 1**

# **Implement Caesar Cypher**

**Aim**: Implement Caesar cipher in cpp

**Theory**: The caesar cipher is traditional way of cryptographyIt is a type of substitution method In this type each letter of a given text is replaced by a letter with a fixed number of positions down the alphabet.In which we shift a letter by 3 shifts.

IMG_256

(Encryption Phase with shift n)

IMG_256

(Decryption Phase with shift n)

Text : ABCDEFGHIJKLMNOPQRSTUVWXYZ

Shift: 23

Cipher: XYZABCDEFGHIJKLMNOPQRSTUVW

**Code**:

#include<bits/stdc++.h>

#include<string.h>

#include<fstream>

using namespace std;

string encrypt(string text, int s)

{

    string result = "";

    for (int i=0;i<text.length();i++)

    {

      if(text[i] == ' ' || text[i] == '\n')

         continue;

        else if (isupper(text[i]))

            result += char(int(text[i]+s-65)%26 +65);

        else

            result += toupper(char(int(text[i]+s-97)%26 +97));

    }

    return result;

}

string decrypt(string cipher,int s)

{

    string result = "";

    for(int i=0;i<cipher.size();i++)

   {

      if(cipher[i] == ' ' || cipher[i] == '\n')

         continue;

        else if (isupper(cipher[i]))

            result += char((int(cipher[i]-s-65)+26)%26 +65);

        else

            result += char((int(cipher[i]-s-97)+26)%26 +97);

    }

    return result;

}

int main()

{

   int choice;

   int datachoice;

   string sample;

   int shift;

   while(1)

   {

      cout << "Caesar Cipher\n 1. Encryption \n 2. Decryption\n 3. Exit\nEnter Choice: ";

      cin>>choice;

      if(choice>2)

         break;

      switch(choice)

      {

         case 1:

            cout << "Data is from\n 1. Manual Entering \n 2. File \nEnter Choice: ";

            cin>>datachoice;

            if(datachoice == 1)

            {

               cout<<"Enter data to be Encrypted:\n";

               cin.ignore();

               getline(cin,sample);

               cout<<"Enter the shift value: ";

               cin>>shift;

               cout<<"Encrypted String:\n";

               cout<<encrypt(sample,shift)<<endl;;

            }

            else

            {

               cout<<"Enter File Name:\n";

               cin>>sample;

               cout<<"Enter the shift value: ";

               cin>>shift;

               fstream myfile;

               myfile.open(sample.c\_str());

               string str,s;

               if(!myfile.is\_open())

                  cout << "Error while Opening File";

               while(getline(myfile,str))

                  s+=str;

               myfile.close();

               s=encrypt(s,shift);

               myfile.open("CipherText.txt",ios\_base::out);

               if(myfile.is\_open())

                  myfile.write(s.data(),s.size());

               cout<<"File Encrypted\n";

               myfile.close();

            }

            break;

         case 2:

            cout << "Data is from\n 1. Manual Entering \n 2. File \nEnter Choice: ";

            cin>>datachoice;

            if(datachoice == 1)

            {

               cout<<"Enter data to be Decrypted:\n";

               cin.ignore();

               getline(cin,sample);

               cout<<"Enter the shift value: ";

               cin>>shift;

               cout<<"Decrypted String:\n";

               cout<<decrypt(sample,shift)<<endl;;

            }

            else

            {

               cout<<"Enter File Name:\n";

               cin>>sample;

               cout<<"Enter the shift value: ";

               cin>>shift;

               fstream myfile;

               myfile.open(sample.c\_str());

               string str,s;

               if(!myfile.is\_open())

                  cout << "Error while Opening File";

               while(getline(myfile,str))

                  s+=str;

               myfile.close();

               s=decrypt(s,shift);

               myfile.open("PlainText.txt",ios\_base::out);

               if(myfile.is\_open())

                  myfile.write(s.data(),s.size());

               cout<<"File Decrypted\n";

               myfile.close();

            }

            break;

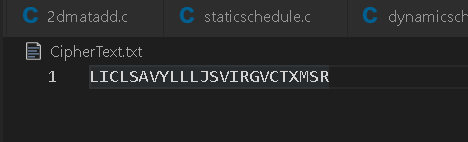
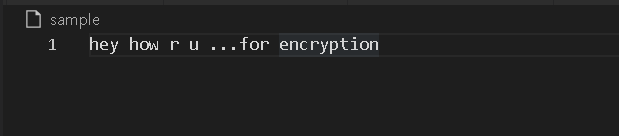
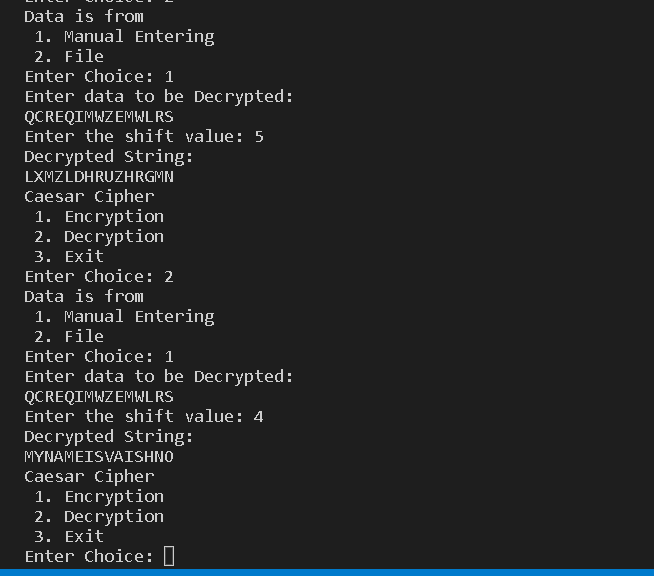
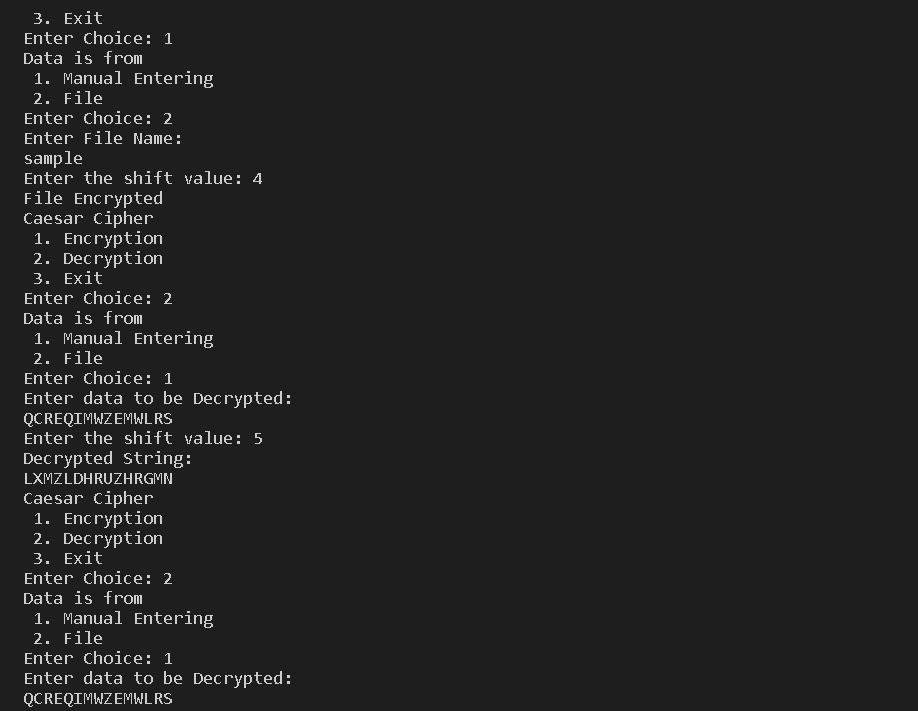
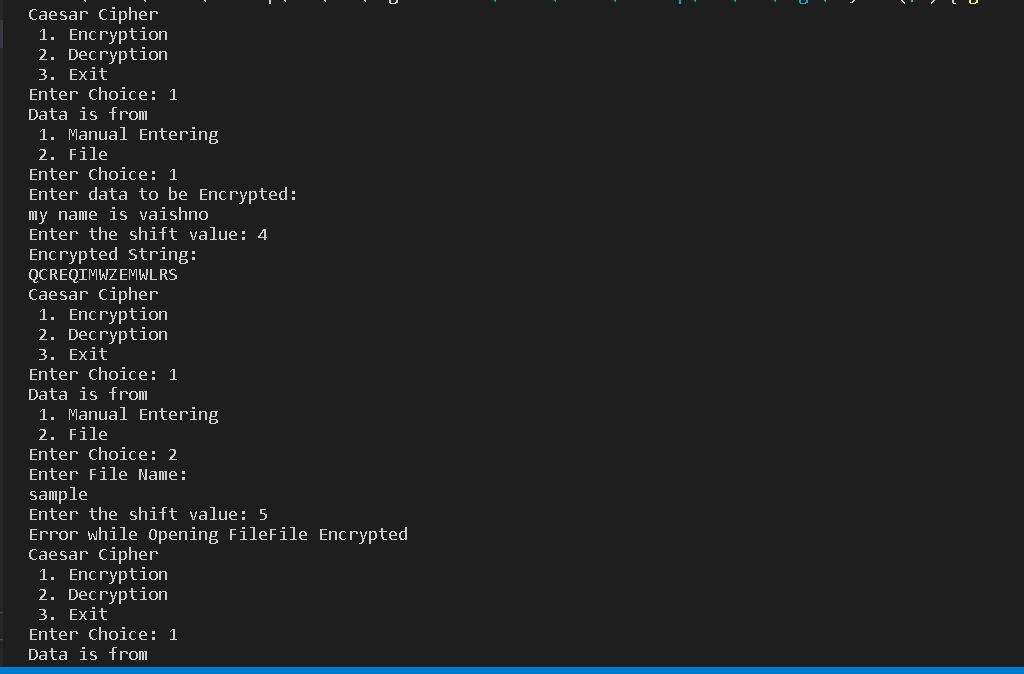
      }

   }

   return 0;

}

Output:



Conclusion:

So Caesar Cypher is easiest but not so efficient because it is so easy to crack using brute force.