

**Experiment No. 1**

**Title: Substitution Cipher**

**Batch:A3** **Roll No.:16010420061** **Experiment No.:1**

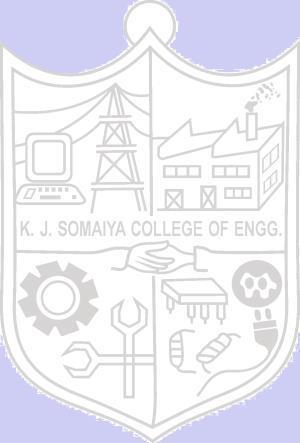
**Aim:** To implement substitution ciphers – Affine and Vigenere cipher.

**Resources needed:** Windows/Linux.

**Theory**

**Pre Lab/ Prior Concepts:**

**Symmetric-key algorithms** are a class [of algorithms for cryptography that](http://en.wikipedia.org/wiki/Algorithm) use the same cryptographic keys for both encryption of plaintext and decryption of cipher text. The keys may be identical or there may be a simple transformation to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link. This requirement that both parties have access to the secret key is one of the main drawbacks of symmetric key encryption, in comparison to public-key encryption. Symmetric-key encryption can use either stream ciphers or block ciphers. Transposition Cipher is block cipher. Ancient cryptographic systems are classified as: Substitution and Permutation Ciphers.

**Simple Substitution Cipher**

A substitution cipher replaces one symbol with another. Letters of plaintext are replaced by other letters or by numbers or symbols. In a particularly simple implementation of a simple substitution cipher, the message is encrypted by substituting the letter of the alphabet n places ahead of the current letter. For example, with n = 3, the substitution which acts as the key

plaintext: a b c d e f g h i j k l m n o p q r s t u v w x y z

ciphertext: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

The convention is plaintext will be in lowercase and the cipher text will be in uppercase. In this example, the key could be stated more succinctly as “3” since the amount of the shift is the key. Using the key of 3, we can encrypt the plaintext message: “fourscoreandsevenyearsago” by looking up each letter in the plaintext row and substituting the corresponding letter in the ciphertext row or by simply replacing each letter by the letter that is three positions ahead of it in the alphabet. In this particular example, the resulting cipher text is IRXUVFRUHDAGVHYHABHDUVDIR

To decrypt, we simply look up the ciphertext letter in the ciphertext row and replace it with the corresponding letter in the plaintext row, or simply shift each ciphertext letter backward by three. The simple substitution with a shift of three is known as the Caesar’s cipher because it was reputedly used with success by Julius Caesar.

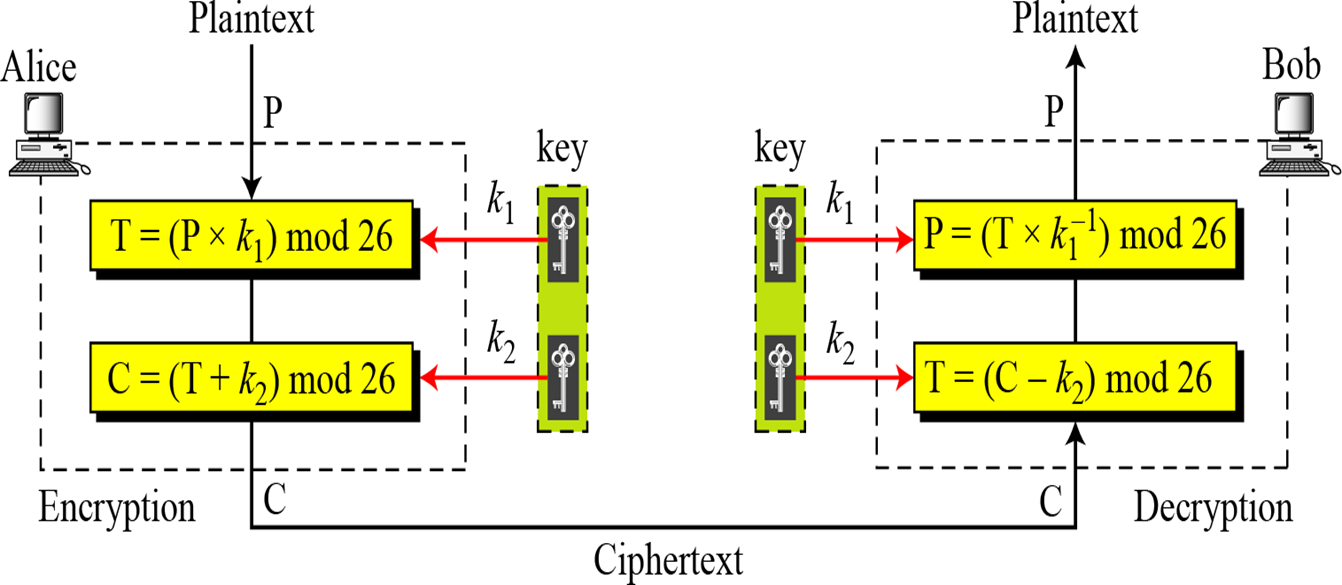
Substitution ciphers are classified as monoalphabetic and polyalphabetic substitution cipher. In monoalphabetic substitution cipher each occurrence of character is encrypted by same substitute character. In Polyalphabetic substitution cipher each occurrence of a character may have a different substitute due to variable Key.

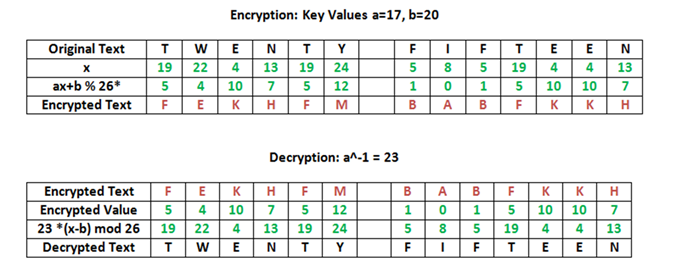
**AFFINE CIPHER**

The Affine cipher is a type of monoalphabetic substitution cipher which uses a combination of Additive and Multiplicative Ciphers.  Each letter is enciphered with the function (ax + b) mod 26, where b is the magnitude of the shift. The encryption function for a single letter is C=(ax + b) mod m where 1≤a≤m, 1≤b≤m{\displaystyle {\mbox{E}}(x)=(ax+b){\bmod {m}},}

where modulus m is the size of the alphabet and a and b are the keys of the cipher. The value a must be chosen such that a and m are coprime. The decryption function is P = a-1(c-b) mod m, where a−1 is the [modular multiplicative inverse](https://en.wikipedia.org/wiki/Modular_multiplicative_inverse) of a i.e., it satisfies the equation a a-1 = 1 mod m.

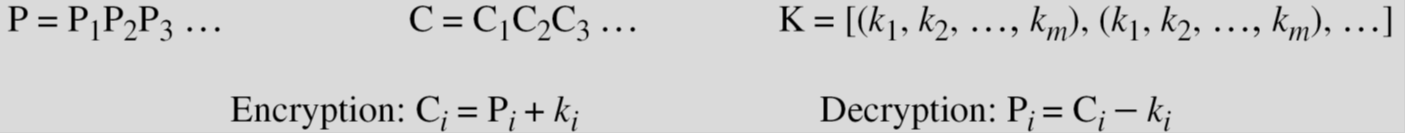
{\displaystyle 1=aa^{-1}{\bmod {m}}.}



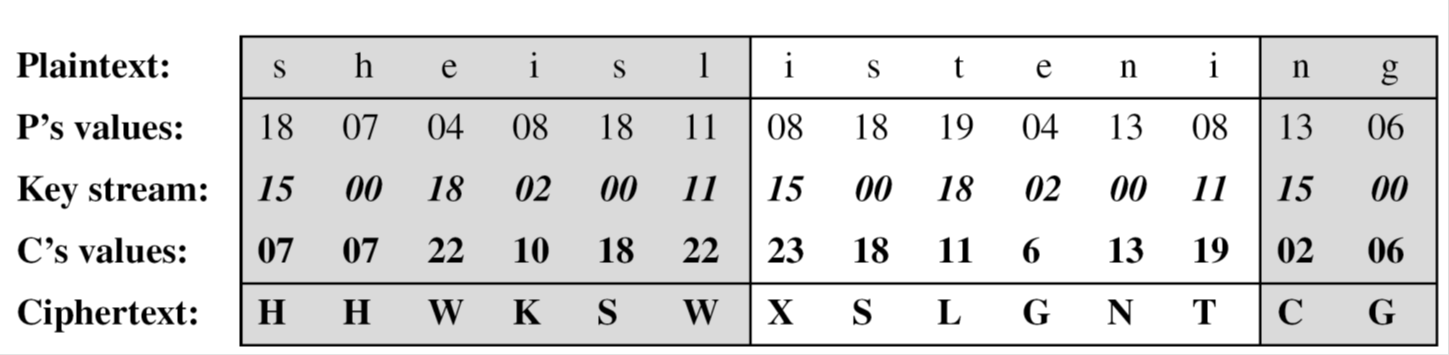
****

**Vigenere Cipher**

Vigenere cipher is a polyalphabetic substitution cipher where each occurrence of a character may have a different substitute due to variable. A set of related monoalphabetic substitution rules are used. A key determines which rule to be used. The relationship between a character in the plaintext to a character in the cipher text is one-to-many.



We can encrypt the message “She is listening” using the 6-character keyword “PASCAL”.

****

**Activity:**

Implement the following substitution ciphers:

1. Affine Cipher
2. Vigenere Cipher

**Implementation:**

The program should have encryption function and decryption function for each cipher. Function should take message and a key as input from the user and display the expected output.

**Results:** (Program with output as per the format)

**Affine Cipher:**

#include<bits/stdc++.h>

using namespace std;

string encryptMessage(string msg,int k1, int k2)

{

    string cipher = "";

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

    {

        if(msg[i]!=' ')

            cipher = cipher + (char) ((((k1 \* (msg[i]-'A') ) + k2) % 26) + 'A');

        else

            cipher += msg[i];

    }

    return cipher;

}

string decryptCipher(string ciphertext,int k1,int k2)

{

    string msg = "";

    int a\_inv = 0;

    int flag = 0;

    for (int i = 0; i < 26; i++)

    {

        flag = (k1 \* i) % 26;

        if (flag == 1)

        {

            a\_inv = i;

        }

    }

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

    {

        if(ciphertext[i]!=' ')

            msg = msg +

                    (char) (((a\_inv \* ((ciphertext[i]+'A' - k2)) % 26)) + 'A');

        else

            msg += ciphertext[i];

    }

    return msg;

}

int main(void)

{

    int k1,k2,dk1,dk2,choice;

    string msg,ciphertext,cipherText,decryptText;

    cout << "Enter 1 for Encryption and 2 for Decryption ";

    cin >> choice;

    switch (choice)

    {

    case 1:

        cout << "Enter the message to be encrypted: ";

        cin >> msg;

        cout << "Enter key1,key2: ";

        cin >> k1 >> k2;

        cipherText = encryptMessage(msg,k1,k2);

        cout << "Shhh! Your Encrypted Message is : " << cipherText;

        break;

    case 2:

        cout << "\nEnter the message to be decrypted: ";

        cin >> ciphertext;

        cout << "Enter key1,key2: ";

        cin >> dk1 >> dk2;

        decryptText = decryptCipher(ciphertext,dk1,dk2);

        cout << "\n And the Decrypted Message is: " << decryptText;

        break;

    default:

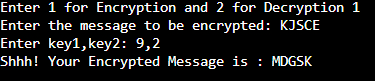
        cout << "Enter the correct choice please!";

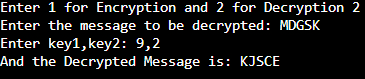
        break;

    }

}

**Output:**

****

****

**Vigenere Cipher:**

#include<bits/stdc++.h>

using namespace std;

string generateKey(string plaintext, string key)

{

    int x = plaintext.size();

    for (int i = 0; ; i++)

    {

        if (x == i)

            i = 0;

        if (key.size() == plaintext.size())

            break;

        key.push\_back(key[i]);

    }

    return key;

}

string encryption(string plaintext, string key)

{

    string cipher\_text;

    for (int i = 0; i < plaintext.size(); i++)

    {

        char x = (plaintext[i] + key[i]) %26;

        x += 'A';

        cipher\_text.push\_back(x);

    }

    return cipher\_text;

}

string decryption(string ciphertext, string key)

{

    string orig\_text;

    for (int i = 0 ; i < ciphertext.size(); i++)

    {

        char x = (ciphertext[i] - key[i] + 26) %26;

        x += 'A';

        orig\_text.push\_back(x);

    }

    return orig\_text;

}

int main()

{

    string keyword,keyword1;

    string ciphertext,plaintext;

    cout << "Enter the plain text: ";

    cin >> plaintext;

    cout << "Enter the keyword: ";

    cin >> keyword;

    string key = generateKey(plaintext, keyword);

    string cipher\_text = encryption(plaintext, key);

    cout << "Shh! Your Ciphertext is : " << cipher\_text << "\n";

    cout << "Enter the Cipher text: ";

    cin >> ciphertext;

    cout << "Enter the keyword: ";

    cin >> keyword1;

    string key1 = generateKey(ciphertext, keyword1);

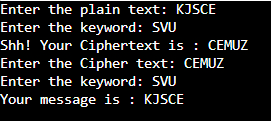
    string decrypt\_text = decryption(ciphertext, key1);

    cout << "Your message is : " << decrypt\_text;

    return 0;

}

**Output:**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Questions:**

1. Write down the flaws of Affine cipher and Vigenere Cipher:

Flaws of Affine Cipher:

1. The cipher's primary weakness comes from the fact that if the cryptanalyst can discover (by means of frequency analysis, brute force, guessing or otherwise) the plaintext of two ciphertext characters then the key can be obtained by solving a simultaneous equation. Since we know key1 (as in the code above) and 26 are relatively prime this can be used to rapidly discard many "false" keys in an automated system.
2. The possible combinations for affine ciphers are only 12\*26 = 312 which is very less and can be easily brute forced and cracked.

Flaws of Vigenere Cipher:

1. The primary weakness of the Vigenere cipher is the repeating nature of its key. If a cryptanalyst correctly guesses the length of the key, then the ciphertext can be treated as interwoven Caesar ciphers, which, individually, can be easily broken.

**Outcomes:**

# CO 1 - Describe the basics of Information Security

**Conclusion:**

Implemented Affine Cipher and Vigenere Cipher successfully.

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**

**References: Books/ Journals/ Websites:**

1. Behrouz A. Forouzan, “Cryptography and Network Security”, Tata McGraw Hill
2. William Stalling, “Cryptography and Network Security”, Prentice Hall