Experiment No. 1

Title: Substitution Cipher

Batch:A2 Roll No.:16010421063 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 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.

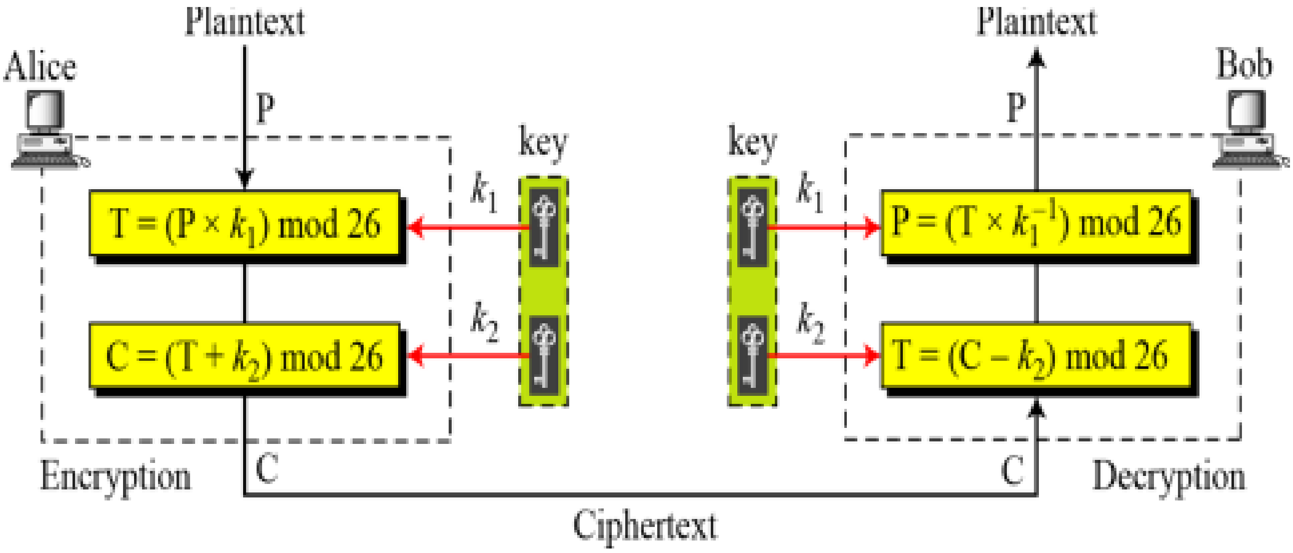
AFFINE CIPHER

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)

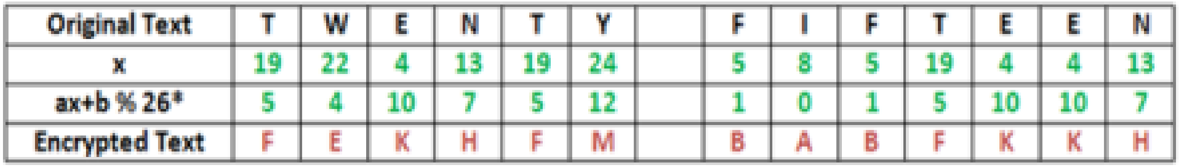
mod 26, where b is the magnitude of the shift. The encryption function for a single letter is

C=(ax + b) mod m where 1≤a≤m, 1≤b≤m

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 of a i.e., it satisfies the equation a a^{-1} = 1 mod m.

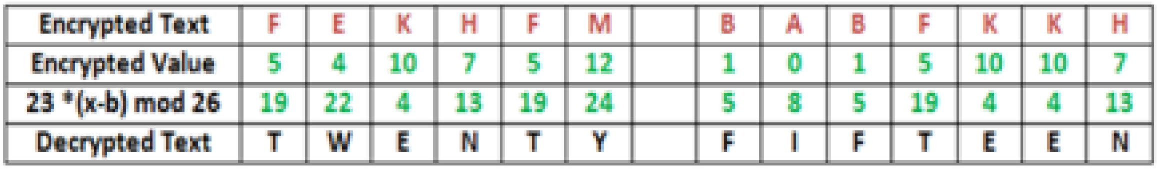


Encryption: Key Values a=17, b=20



Original Text	T	W	E	N	T	Y		F	I	F	T	E	E	N
x	19	22	4	13	19	24		5	8	5	19	4	4	13
ax+b % 26*	5	4	10	7	5	12		1	0	1	5	10	10	7
Encrypted Text	F	E	K	H	F	M		B	A	B	F	K	K	H

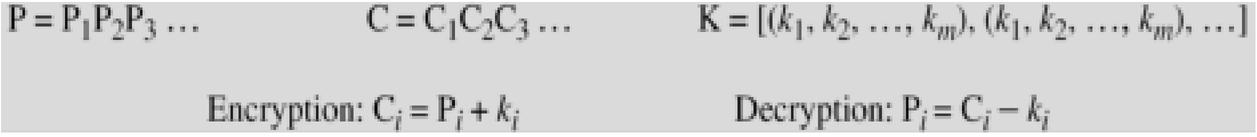
Decryption: a^-1 = 23



Encrypted Text	F	E	K	H	F	M		B	A	B	F	K	K	H
Encrypted Value	5	4	10	7	5	12		1	0	1	5	10	10	7
23 *(x-b) mod 26	19	22	4	13	19	24		5	8	5	19	4	4	13
Decrypted Text	T	W	E	N	T	Y		F	I	F	T	E	E	N

Vigenere Cipher

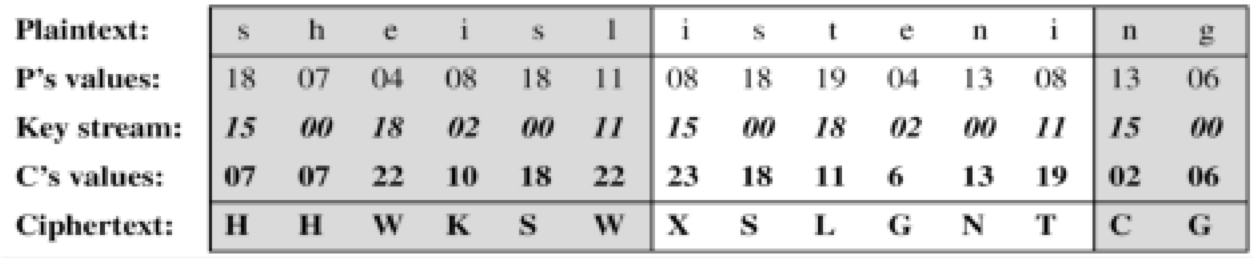
Vigenere cipher is a polyalphabetic substitution cipher where each occurrence of a character may have a different substitute due to variable. A set of related monoalphabetic substitution rules are used. A key determines which rule to be used. The relationship between a character in the plaintext to a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



$P = P_1P_2P_3 \ldots$ $C = C_1C_2C_3 \ldots$ $K = [(k_1, k_2, \ldots, k_m), (k_1, k_2, \ldots, k_m), \ldots]$

Encryption: $C_i = P_i + k_i$ Decryption: $P_i = C_i - k_i$

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



Plaintext:	s	h	e	i	s	l	i	s	t	e	n	i	n	g
P’s values:	18	07	04	08	18	11	08	18	19	04	13	08	13	06
Key stream:	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***
C’s values:	**07**	**07**	**22**	**10**	**18**	**22**	**23**	**18**	**11**	**6**	**13**	**19**	**02**	**06**
Ciphertext:	**H**	**H**	**W**	**K**	**S**	**W**	**X**	**S**	**L**	**G**	**N**	**T**	**C**	**G**

Activity:

Implement the following substitution ciphers:

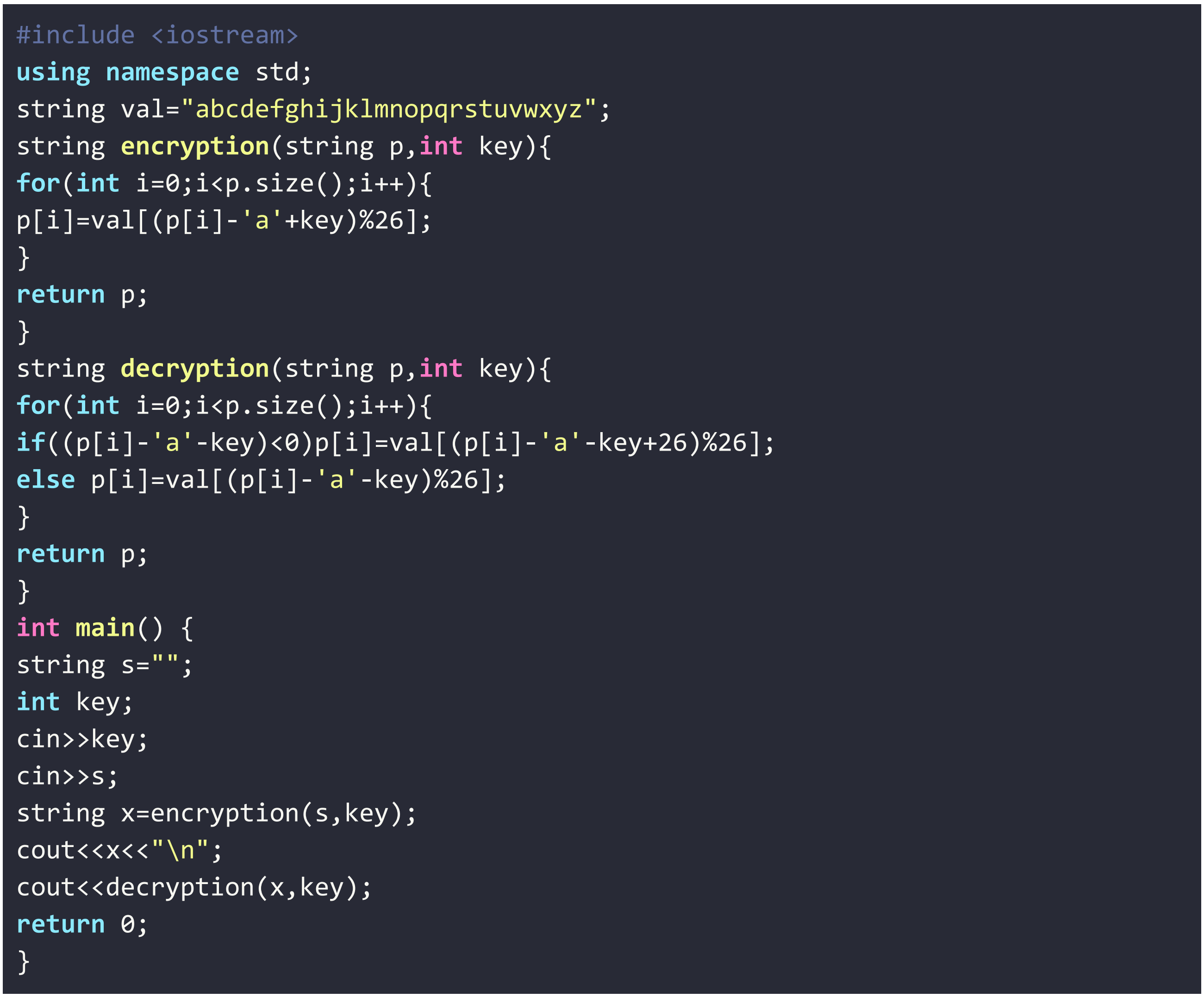
1. Affine Cipher
2. Vigenere Cipher

Implementation:

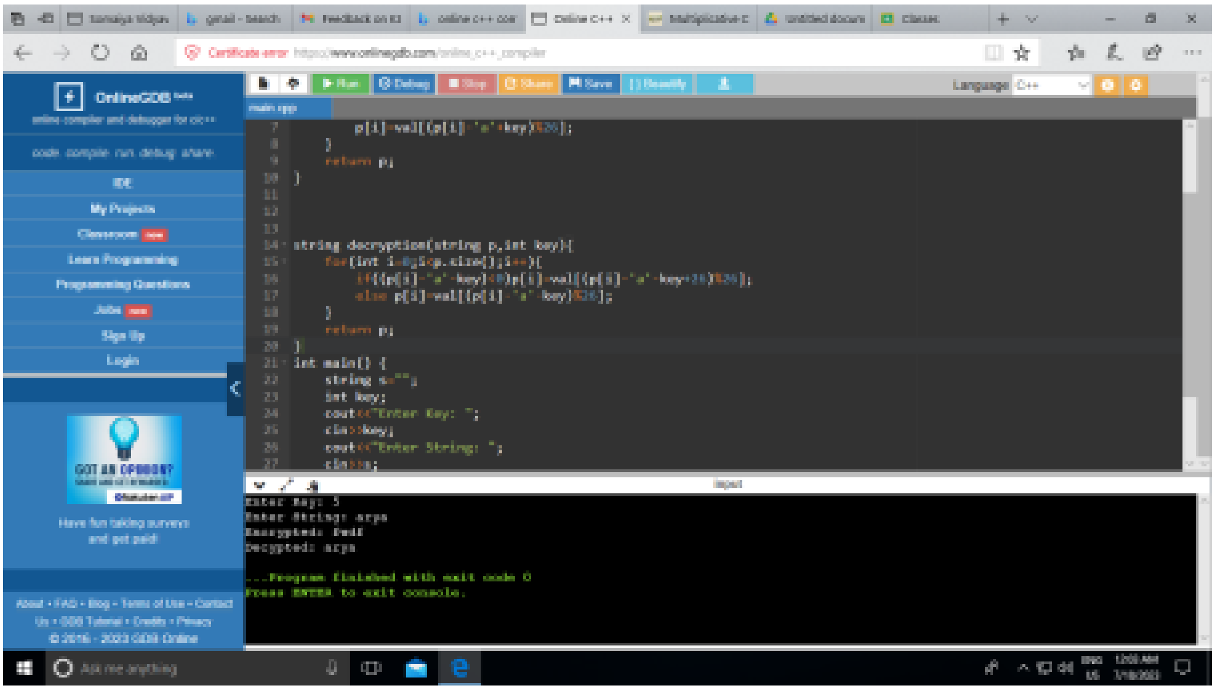
The program should have encryption function and decryption function for each cipher. Function should take message and a key as input from the user and display the expected output.

Results: (Program with output as per the format)

Additive Cipher



```
#include <iostream>
using namespace std;
string val="abcdefghijklmnopqrstuvwxyz";
string encryption(string p,int key){
for(int i=0;i<p.size();i++){
p[i]=val[(p[i]-'a'+key)%26];
}
return p;
}
string decryption(string p,int key){
for(int i=0;i<p.size();i++){
if((p[i]-'a'-key)<0)p[i]=val[(p[i]-'a'-key+26)%26];
else p[i]=val[(p[i]-'a'-key)%26];
}
return p;
}
int main() {
string s="";
int key;
cin>>key;
cin>>s;
string x=encryption(s,key);
cout<<x<<"\n";
cout<<decryption(x,key);
return 0;
}
```



```
        p[i]=val[(p[i]-'a'+key)%26];
    }
    return p;
}



string decryption(string p,int key){
    for(int i=0;i<p.size();i++){
        if((p[i]-'a'-key)<0)p[i]=val[(p[i]-'a'-key+26)%26];
        else p[i]=val[(p[i]-'a'-key)%26];
    }
    return p;
}
int main() {
    string s="";
    int key;
    cout<<"Enter Key: ";
    cin>>key;
    cout<<"Enter String: ";
    cin>>s;
```

```
Enter Key: 3
Enter String: arya
Encrypted: Dwdf
Decrypted: arya

...Program finished with exit code 0
Press ENTER to exit console.
```

With Socket

server.py

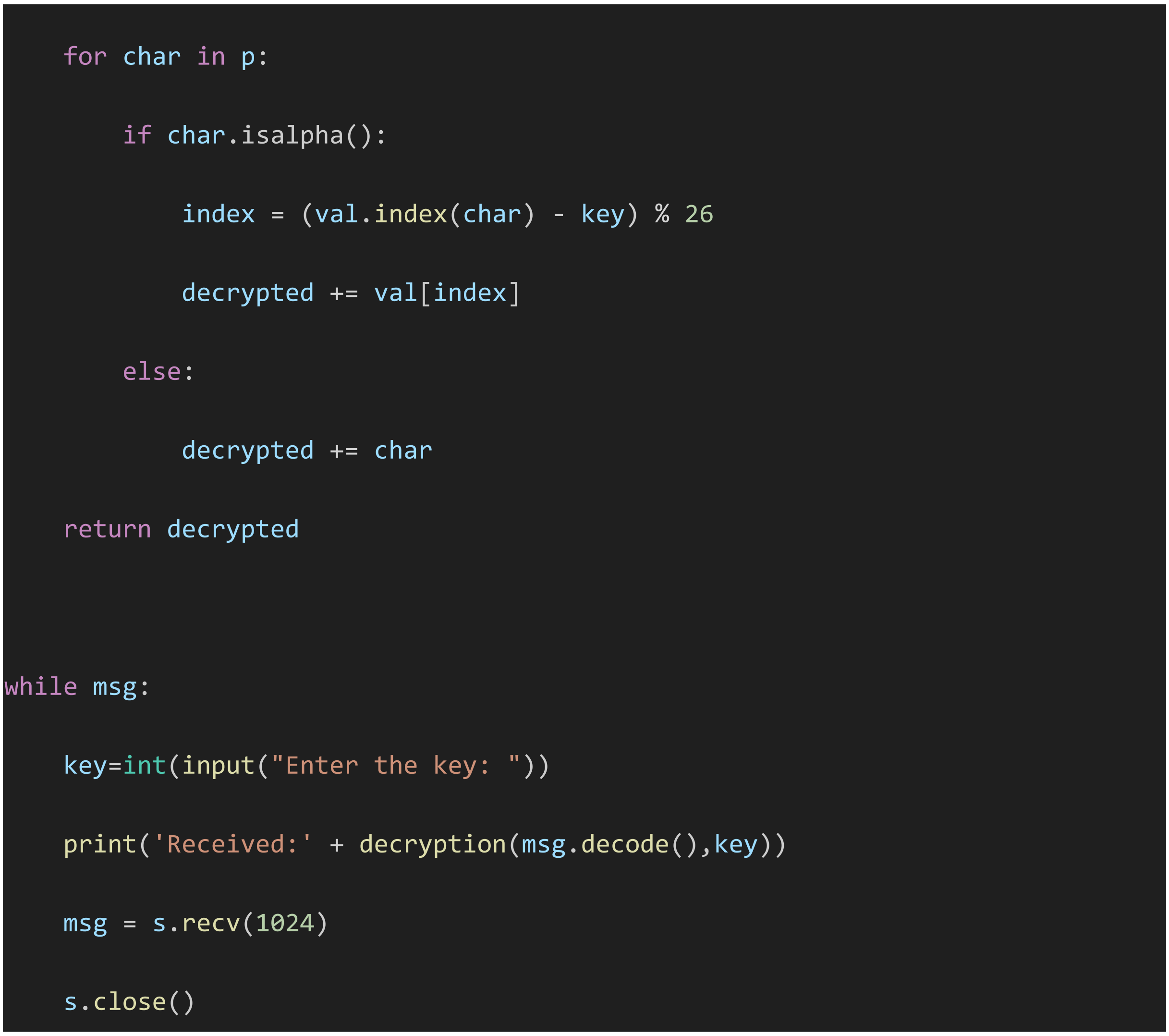
```
import socket
```

```
s = socket.socket(socket.AF_INET,
                  socket.SOCK_STREAM)
print ("Socket successfully created")

port = 12345

s.bind(('', port))
print ("socket binded to %s" %(port))

s.listen(1)
print ("socket is listening")

def encryption(p, key):
    val = "abcdefghijklmnopqrstuvwxyz"
    encrypted = ""
    for char in p:
        if char.isalpha():
            index = (val.index(char) + key) % 26
            encrypted += val[index]
        else:
            encrypted += char
    return encrypted

c, addr = s.accept()
print(str(addr))
```

```
txt=input("Enter Text to send or 'C' to quit: ")
key=int(input("Enter Key: "))
txt=encryption(txt,key)
c.sendall(txt.encode())
c.close()
```

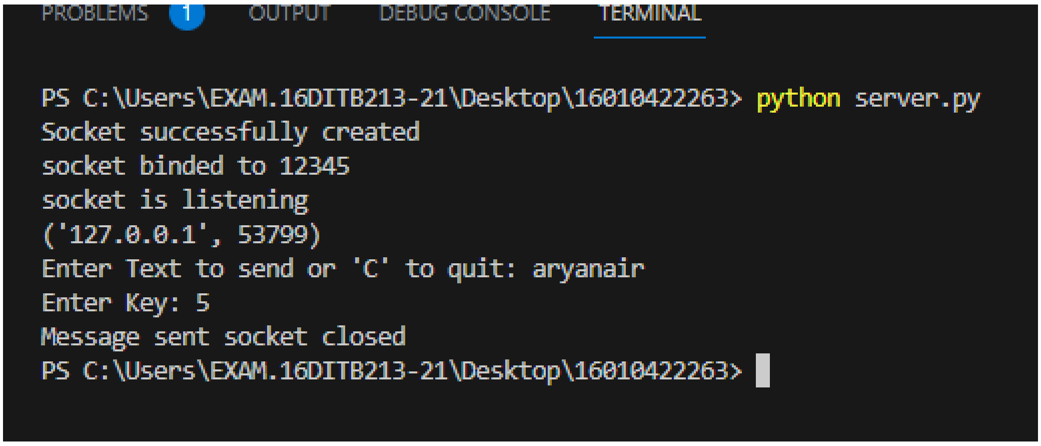
Client.py

```

import socket

# take the server name and port name

host = 'local host'
port = 12345

s = socket.socket(socket.AF_INET,
                  socket.SOCK_STREAM)

s.connect(('127.0.0.1', port))

msg = s.recv(1024)

def decryption(p, key):
    val = "abcdefghijklmnopqrstuvwxyz"
    decrypted = ""
```



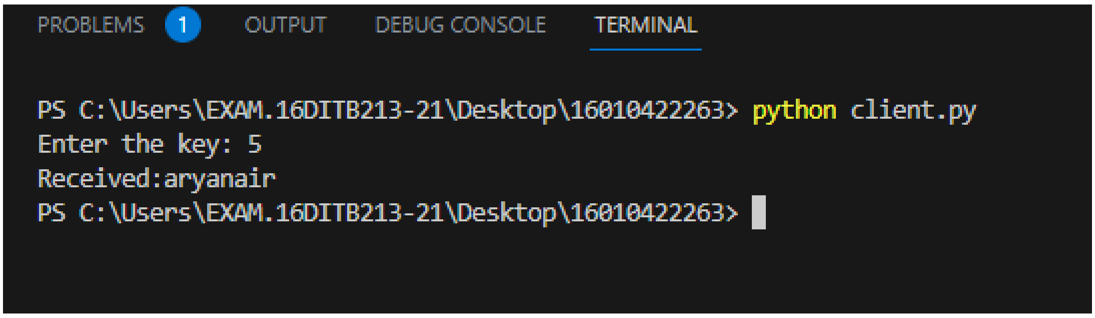
```
    for char in p:

        if char.isalpha():

            index = (val.index(char) - key) % 26

            decrypted += val[index]

        else:

            decrypted += char

    return decrypted



while msg:

    key=int(input("Enter the key: "))

    print('Received:' + decryption(msg.decode(),key))

    msg = s.recv(1024)

    s.close()
```

Server



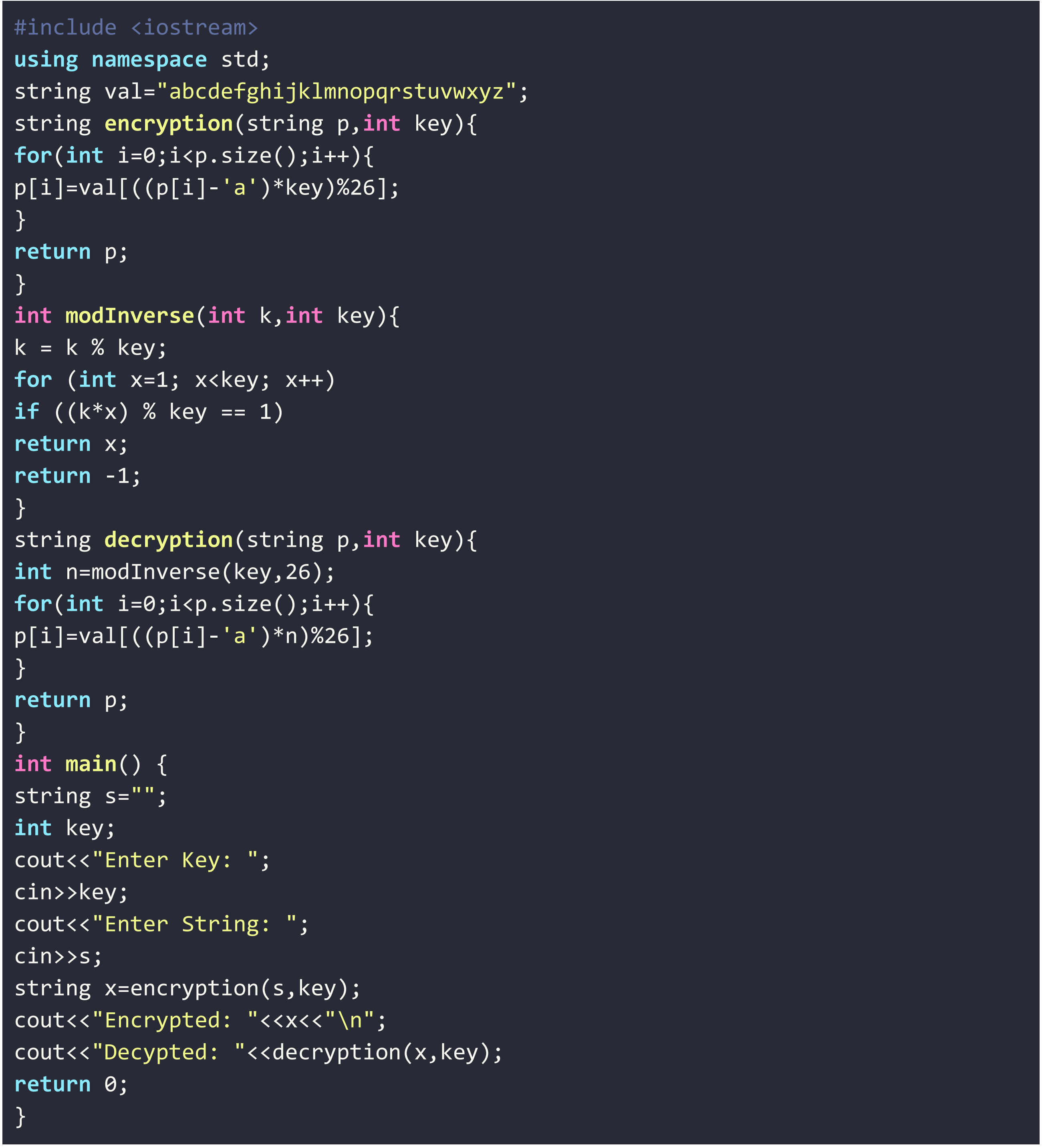
```
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263> python server.py
Socket successfully created
socket binded to 12345
socket is listening
('127.0.0.1', 53799)
Enter Text to send or 'C' to quit: aryanair
Enter Key: 5
Message sent socket closed
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263>
```

Client

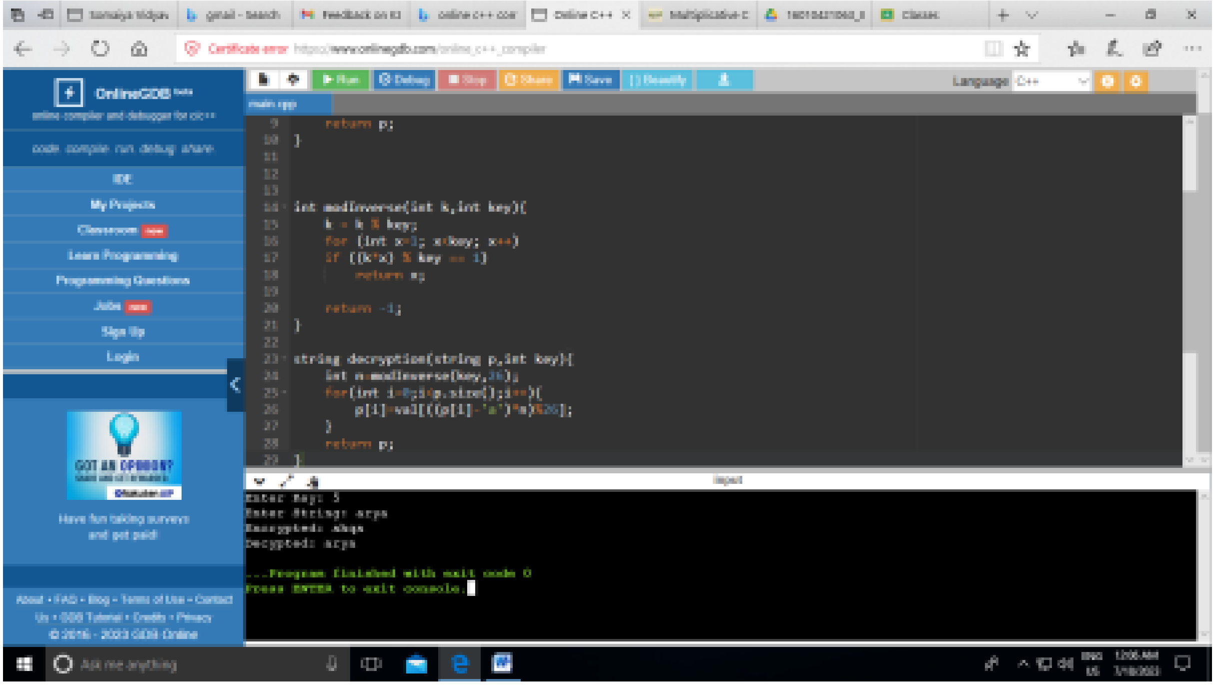


```
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263> python client.py
Enter the key: 5
Received:aryanair
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263>
```

Multiplicative Cipher



```cpp
#include <iostream>
using namespace std;
string val="abcdefghijklmnopqrstuvwxyz";
string encryption(string p,int key){
for(int i=0;i<p.size();i++){
p[i]=val[((p[i]-'a')*key)%26];
}
return p;
}
int modInverse(int k,int key){
k = k % key;
for (int x=1; x<key; x++)
if ((k*x) % key == 1)
return x;
return -1;
}
string decryption(string p,int key){
int n=modInverse(key,26);
for(int i=0;i<p.size();i++){
p[i]=val[((p[i]-'a')*n)%26];
}
return p;
}
int main() {
string s="";
int key;
cout<<"Enter Key: ";
cin>>key;
cout<<"Enter String: ";
cin>>s;
string x=encryption(s,key);
cout<<"Encrypted: "<<x<<"\n";
cout<<"Decypted: "<<decryption(x,key);
return 0;
}
```



```
    return p;
}



int modInverse(int k,int key){
    k = k % key;
    for (int x=1; x<key; x++)
    if ((k*x) % key == 1)
        return x;

    return -1;
}

string decryption(string p,int key){
    int n=modInverse(key,26);
    for(int i=0;i<p.size();i++){
        p[i]=val[((p[i]-'a')*n)%26];
    }
    return p;
}
```

```
Enter Key: 5
Enter String: arps
Encrypted: ahqs
Decrypted: arps

...Program finished with exit code 0
Press ENTER to exit console.
```

With Socket
Server

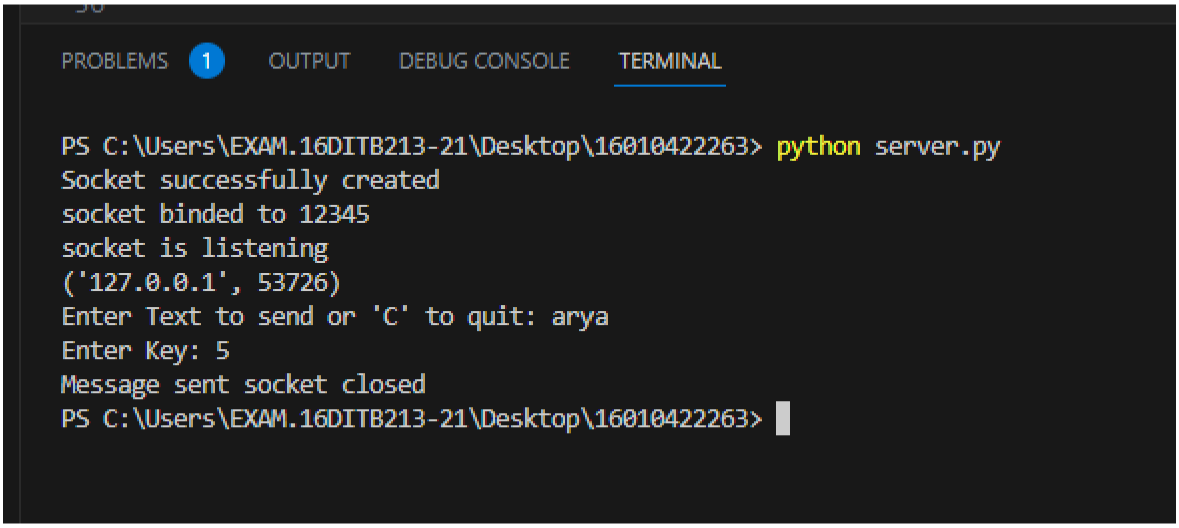
```
import socket

s = socket.socket(socket.AF_INET,
                  socket.SOCK_STREAM)
print ("Socket successfully created")

port = 12345

s.bind(('', port))
print ("socket binded to %s" %(port))

s.listen(1)
print ("socket is listening")

def encryption(p, key):
    val = "abcdefghijklmnopqrstuvwxyz"
    encrypted = ""
    for char in p:
        if char.isalpha():
            index = (val.index(char) * key) % 26
            encrypted += val[index]
        else:
            encrypted += char
    return encrypted


c, addr = s.accept()
print(str(addr))
txt=input("Enter Text to send or 'C' to quit: ")
```

```
key=int(input("Enter Key: "))
txt=encryption(txt,key)
c.sendall(txt.encode())
c.close()
print("Message sent socket closed")
```

Client

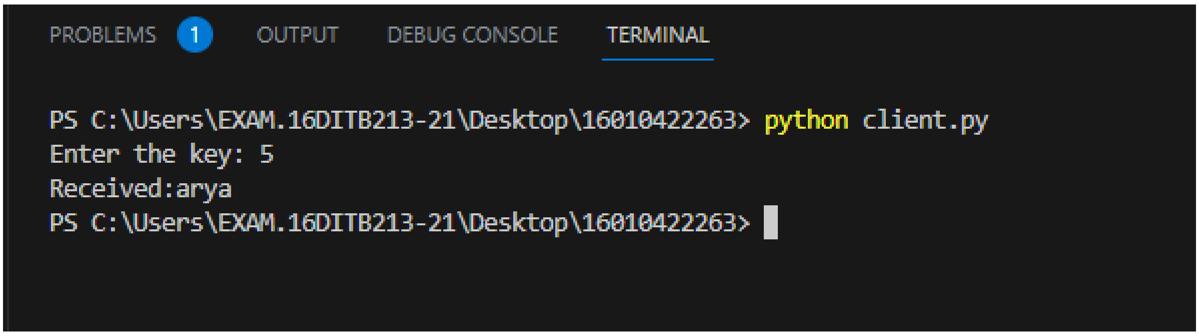
```
import socket

# take the server name and port name

host = 'local host'
port = 12345

s = socket.socket(socket.AF_INET,
                  socket.SOCK_STREAM)

s.connect(('127.0.0.1', port))

msg = s.recv(1024)

def mod_inverse(k, key):
    k = k % key
    for x in range(1, key):
        if (k * x) % key == 1:
            return x
    return -1

def decryption(p, key):
    val = "abcdefghijklmnopqrstuvwxyz"
    n = mod_inverse(key, 26)
    decrypted = ""
    for char in p:
        if char.isalpha():
            index = (val.index(char) * n) % 26
            decrypted += val[index]
        else:
            decrypted += char
    return decrypted

while msg:
    key=int(input("Enter the key: "))
    print('Received:' + decryption(msg.decode(),key))
    msg = s.recv(1024)
    s.close()
```

Server terminal



```
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263> python server.py
Socket successfully created
socket binded to 12345
socket is listening
('127.0.0.1', 53726)
Enter Text to send or 'C' to quit: arya
Enter Key: 5
Message sent socket closed
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263>
```

Client terminal



```
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263> python client.py
Enter the key: 5
Received:arya
PS C:\Users\EXAM.16DITB213-21\Desktop\16010422263>
```

Questions:

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

Affine Cipher:

- The affine cipher is **slightly more complicated than the Caesar cipher**. • It is a type of simple substitution cipher that is **very easy to crack**. • Affine ciphers can be cracked if any 2 characters are known. As hand ciphers, affine ciphers are too complex to be practically applied without the aid of an explicit lookup table.
- While one could construct an encryption table using an affine map, doing so would not seem to offer any particular advantage over other methods actually used in practice.

Vigenere Cipher

- It is a type of simple substitution cipher that is **very easy to crack**.
- Vigenere ciphers can be cracked if any 2 characters are known.
- Vigenere ciphers are too complex to be practically applied without the aid of an explicit lookup table
- The Vignere cipher is **slightly more complicated than the Caesar cipher**.

Outcomes:

CO 1 Describe the basics of Information Security

Conclusion:
Successfully understood substitution cipher and implemented additive and multiplicative cipher

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

(A Constituent College of Somaiya Vidyavihar University)