{

Question 1:-Ceaser Cipher

```
import java.util.*;

class ceaser_cipher

{

        public static void main(String args[])throws Exception

        {

                Scanner sc=new Scanner(System.in);

                System.out.print("Enter text you want to encrypt:");

                String input=sc.nextLine();

                int key=3;

                conversion c=new conversion();

                String encrypt=c.convert((key*-1),input);

                System.out.println("encrypted message:"+encrypt);

                String decrypt=c.convert(key,encrypt);

                System.out.println("decrypted message:"+decrypt);

        }

}

class conversion

{

        char temp,t;

        int asci=0;

        public String convert(int key,String inputmsg)

        {
}

String op="";
int min=0,max=0,flag=0;
for(int i=0;i<inputmsg.length();i++)
{
        temp=inputmsg.charAt(i);
        asci=temp+key;
        if (temp>=97&&temp<=122)
        {

                min=97;
                max=122;
                flag=1;
        }
        else if(temp>=48&&temp<=57)
        {
                min=48;
                max=57;
                flag=1;
        }
        else if(temp>=65&&temp<=90)
        {
                min=65;
                max=90;
                flag=1;
        }
```

```
else
        flag=0;

if(flag==1)
{
        if(asci>max)
        {
                int rem=asci-max;
                t=(char)((min-1)+rem);
        }
        else if(asci<min)
        {
                int rem=min-asci;
                t=(char)((max+1)-rem);
        }
        else
        {
                t=(char)(asci);
        }
        op=op+t;
}
else
{
        op=op+temp;
}
```

```
				}
				return op;
			}
}
```

Output:-

```
D:\sem_5\network_security>java ceaser_cipher
Enter text you want to encrypt:vivek vyas123
encrypted message:sfsbh svxp890
decrypted message:vivek vyas123
```

Question 2:-substituion cipher

```
 import java.util.*;
class substitue_cipher
{
			public static void main(String args[])throws Exception
			{
				Scanner sc=new Scanner(System.in);
				System.out.print("Enter text you want to encrypt:");
				String input=sc.nextLine();
				System.out.print("Enter value of key:");
				int key=sc.nextInt();
				conversion c=new conversion();
```

```
                String encrypt=c.convert((key*-1),input);

                System.out.println("encrypted message:"+encrypt);

                String decrypt=c.convert(key,encrypt);

                System.out.println("decrypted message:"+decrypt);

        }

}

class conversion

{

        char temp,t;

        int asci=0;

        public String convert(int key,String inputmsg)

        {

                String op="";

                int min=0,max=0,flag=0;

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

                {

                        temp=inputmsg.charAt(i);

                        asci=temp+key;

                        if (temp>=97&&temp<=122)

                        {


                                min=97;

                                max=122;

                                flag=1;

                        }
```

```
else if(temp>=48&&temp<=57)
{
        min=48;
        max=57;
        flag=1;
}
else if(temp>=65&&temp<=90)
{
        min=65;
        max=90;
        flag=1;
}
else
        flag=0;

if(flag==1)
{
        if(asci>max)
        {
                int rem=asci-max;
                t=(char)((min-1)+rem);
        }
        else if(asci<min)
        {
                int rem=min-asci;
```

```
                        t=(char)((max+1)-rem);
                    }
                    else
                    {
                        t=(char)(asci);
                    }
                    op=op+t;
                }
                else
                {
                    op=op+temp;
                }
            }
            return op;
        }
}
```

OUTPUT:-

```
D:\sem_5\network_security>java substitue_cipher
Enter text you want to encrypt:vivek vyas123
Enter value of key:5
encrypted message:qdqzf qtvn678
decrypted message:vivek vyas123
```

Question 3:-Transposition Cipher

```
import java.util.*;
class transposition
{
	public static void main(String args[])throws Exception
	{
		Scanner sc=new Scanner(System.in);
		System.out.println("enter key(of unique alphabets):");
		String k=sc.nextLine();
		char[] key=k.toCharArray();
		char[] temp_key=new char[key.length];
		System.arraycopy(key,0,temp_key,0,key.length);
		Arrays.sort(temp_key);
		System.out.print("\nenter string :");
		String t=sc.nextLine();
		char[] str=t.toCharArray();
		for(int i=0;i<str.length;i++)
		{
			if(str[i]==' ')
				str[i]='$';
		}
		int index=0,row;
		if(((str.length)%(key.length))==0)
			row=((str.length)/(key.length));
```

```
else
        row=((str.length)/(key.length))+1;
char[] cipher=new char[(row*(key.length))];
int ci=0;
while(ci<(row*(key.length)))
{
        for(int i=0;i<key.length;i++)
        {
        index=0;
        for(int j=0;j<key.length;j++)
        {
                if(temp_key[i]==key[j])
                {
                        index=j;
                        int l=0;
                        while(l<row)
                        {
                                if(index<str.length)
                                {
                                cipher[ci]=str[index];
                                ci++;
                                l++;
                                index=index+(key.length);
                                }
                                else
```

```
                                        {
                                                  cipher[ci]='!';
                                                  ci++;
                                                  l++;
                                        }
                              }
                              break;
                    }
          }
          }
}
System.out.println("Cipher text:");
for(int i=0;i<cipher.length;i++)
{
          System.out.print(cipher[i]);
}
char[] decipher=new char[cipher.length];
int di=0;
int l=0;
while(di<cipher.length)
{
          for(int i=0;i<key.length;i++)
          {
          index=0;
                    for(int j=0;j<key.length;j++)
                    {
                        if(key[i]==temp_key[j])
                        {
                            index=((j)*row)+l;
                            decipher[di]=cipher[index];
                            if(decipher[di]=='$')
                                decipher[di]=' ';
                            if(decipher[di]=='!')
                                decipher[di]='\0';
                            di++;
                            break;
                        }
                    }
                }
                l++;
            }
            System.out.println("\ndecipher text:");
            for(int i=0;i<cipher.length;i++)
            {
                System.out.print(decipher[i]);
            }
        }
}
```

OUTPUT-

```
D:\sem_5\network_security>java transposition

enter key(of unique alphabets):

mater


enter string :vivek vyas123

Cipher text:

iv2ea!v$1ks!vy3

decipher text:

vivek vyas123
```

Question 4:-One time pad

```
import java.util.*;

class OnePad

{

        public static void main(String args[])throws Exception

        {

                Scanner sc=new Scanner(System.in);

                System.out.println("enter the message:");

                String plain_txt=sc.nextLine();

                char[] pad=new char[plain_txt.length()];

                Random ran = new Random();

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

                {
}

                int asci=ran.nextInt(123);

                if(asci<=127)

                {

                        pad[i]=(char)asci;

                }

                else

                        i--;

        }

        System.out.println(pad);

        char[] msg=plain_txt.toCharArray();

        char[] cipher_txt=new char[msg.length];

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

        {

                cipher_txt[i]=(char)(msg[i]^pad[i]);

        }

        System.out.print("cipher text:");

        System.out.print(cipher_txt);

        char[] original_txt=new char[msg.length];

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

        {

                original_txt[i]=(char)(pad[i]^cipher_txt[i]);

        }

        System.out.print("\noriginal text:");

        System.out.print(original_txt);

}
```

```
}
```

OUTPUT

```
D:\sem_5\network_security>java OnePad

enter the message:

vivek vyas123

yDi-,
MsZWpu

cipher text:-{G+;

`)fBF

original text:vivek vyas123
```

Question 5:- Substitution Client server

----

client_cipher.java:-

```
import java.io.*;

import java.util.*;

import java.net.*;

class client_cipher

{

        public static void main(String args[])throws Exception

        {

                Scanner sc=new Scanner(System.in);

                System.out.print("Enter text you send to server:");

                String input=sc.nextLine();
                File f=new File("key.txt");
                BufferedReader br=new BufferedReader(new FileReader(f));
                Socket s=new Socket("127.0.0.1",1234);
                DataOutputStream dos=new DataOutputStream(s.getOutputStream());
                int key=Integer.parseInt(br.readLine());
                System.out.println("key:"+key);
                conversion c=new conversion();
                String encrypt=c.convert((key*-1),input);
                dos.writeUTF(encrypt);
                br.close();
        }
}
class conversion
{
        char temp,t;
        int asci=0;
        public String convert(int key,String inputmsg)
        {
                String op="";
                int min=0,max=0,flag=0;
                for(int i=0;i<inputmsg.length();i++)
                {
                        temp=inputmsg.charAt(i);
                        asci=temp+key;
                        if (temp>=97&&temp<=122)
```

```
{

        min=97;
        max=122;
        flag=1;
}
else if(temp>=48&&temp<=57)
{
        min=48;
        max=57;
        flag=1;
}
else if(temp>=65&&temp<=90)
{
        min=65;
        max=90;
        flag=1;
}
else
        flag=0;

if(flag==1)
{
        if(asci>max)
        {
```

```
                        int rem=asci-max;
                        t=(char)((min-1)+rem);
                    }
                    else if(asci<min)
                    {
                        int rem=min-asci;
                        t=(char)((max+1)-rem);
                    }
                    else
                    {
                        t=(char)(asci);
                    }
                    op=op+t;
                }
                else
                {
                    op=op+temp;
                }
            }
            return op;
        }
}
```

Server_cipher.java:-

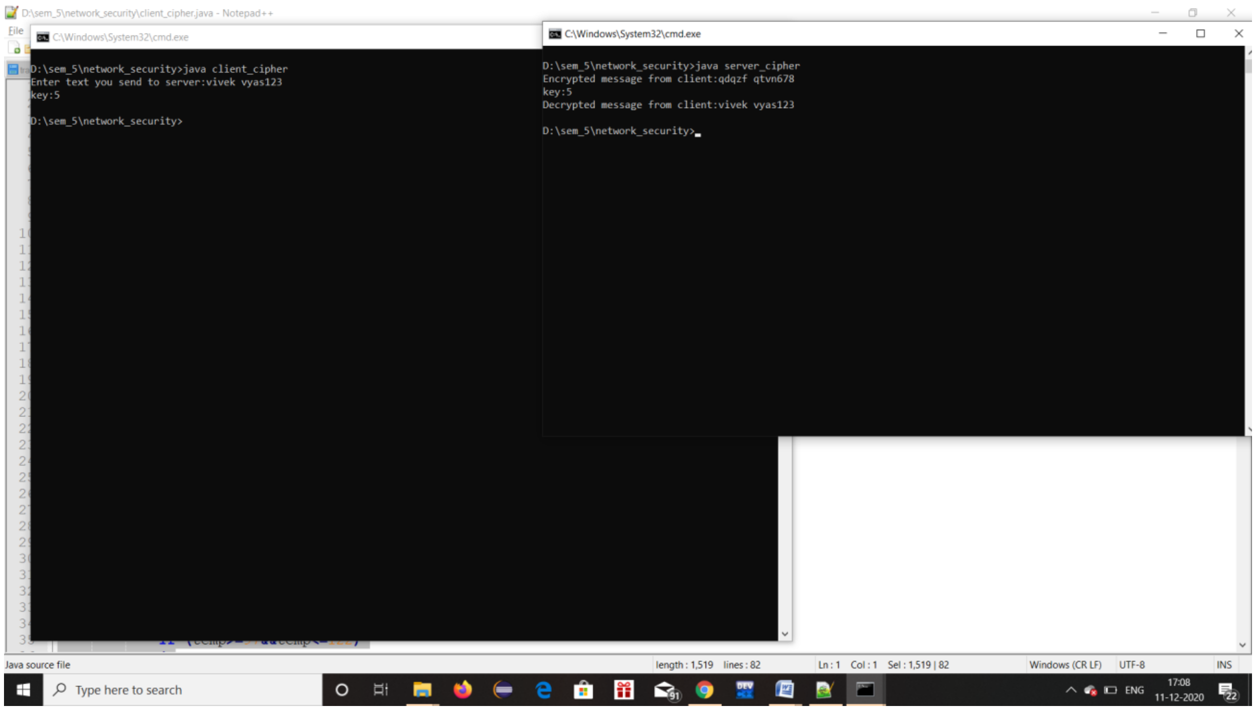
```
import java.io.*;
import java.util.*;
import java.io.*;

import java.net.*;

class server_cipher

{

	public static void main(String args[])throws Exception

	{

		ServerSocket ss=new ServerSocket(1234);

		Socket s=ss.accept();

		DataInputStream dis=new DataInputStream(s.getInputStream());

		String encrypt=dis.readUTF();

		System.out.println("Encrypted message from client:"+encrypt);

		File f=new File("key.txt");

		BufferedReader br=new BufferedReader(new FileReader(f));

		int key=Integer.parseInt(br.readLine());

		System.out.println("key:"+key);

		conversion c=new conversion();

		String decrypt=c.convert(key,encrypt);

		System.out.println("Decrypted message from client:"+decrypt);

	}

}

class conversion

{

	char temp,t;

	int asci=0;

	public String convert(int key,String inputmsg)
```

```
{
        String op="";
        int min=0,max=0,flag=0;
        for(int i=0;i<inputmsg.length();i++)
        {
                temp=inputmsg.charAt(i);
                asci=temp+key;
                if (temp>=97&&temp<=122)
                {

                        min=97;
                        max=122;
                        flag=1;
                }
                else if(temp>=48&&temp<=57)
                {
                        min=48;
                        max=57;
                        flag=1;
                }
                else if(temp>=65&&temp<=90)
                {
                        min=65;
                        max=90;
                        flag=1;
```

```
}
else
        flag=0;

if(flag==1)
{
        if(asci>max)
        {
                int rem=asci-max;
                t=(char)((min-1)+rem);
        }
        else if(asci<min)
        {
                int rem=min-asci;
                t=(char)((max+1)-rem);
        }
        else
        {
                t=(char)(asci);
        }
        op=op+t;
}
else
{
        op=op+temp;
```

```
                    }
                }
                return op;
        }
}
```

OUTPUT:-



```
D:\sem_5\network_security>java client_cipher
Enter text you send to server:vivek vyas123
key:5

D:\sem_5\network_security>
```

```
D:\sem_5\network_security>java server_cipher
Encrypted message from client:qdqzf qtvn678
key:5
Decrypted message from client:vivek vyas123

D:\sem_5\network_security>
```

Question 6:-P-box

```
import java.util.Scanner;
class P_Box{
        public String doEncryption(String s){
```

```
        byte p[]=new byte[8];

        byte pTemp[]=new byte[8];

        pTemp=s.getBytes();

        p[0]=pTemp[4];

        p[1]=pTemp[0];

        p[2]=pTemp[5];

        p[3]=pTemp[7];

        p[4]=pTemp[1];

        p[5]=pTemp[3];

        p[6]=pTemp[2];

        p[7]=pTemp[6];

        return(new String(p));

}

public String doDecryption(String s){

        byte p[]=new byte[8];

        byte pTemp[]=new byte[8];

        pTemp=s.getBytes();

        p[0]=pTemp[1];

        p[1]=pTemp[4];

        p[2]=pTemp[6];

        p[3]=pTemp[5];

        p[4]=pTemp[0];

        p[5]=pTemp[2];

        p[6]=pTemp[7];

        p[7]=pTemp[3];
```

```
                return(new String(p));
        }
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                System.out.println("Enter String(Only 8 Characters) : ");
                String plaintext=sc.nextLine();
                P_Box p_box=new P_Box();
                System.out.println("Encrypted Text : " + p_box.doEncryption(plaintext));
                System.out.println("Decrypted Text : " +
p_box.doDecryption(p_box.doEncryption(plaintext)));
        }
}
```

OUTPUT:-

```
---------------------------------------------------------------------------------------------------------------------
D:\sem_5\network_security>javac P_Box.java


D:\sem_5\network_security>java P_Box
Enter String(Only 8 Characters) :
helloworld
Encrypted Text : ohwrello
Decrypted Text : hellowor
---------------------------------------------------------------------------------------------------------------------
```

Question 7:S-box

```
---------------------------------------------------------------------------------------------------------------------
import java.util.*;
class S_Box{
```

```
char key[][];

Random r;

S_Box(){

        r=new Random();

        int add=r.nextInt(5);

        key=new char[52][2];

        char temp='A',ch;

        for(int i=0;i<key.length;i++,temp++){

                if(temp<='Z' && temp>='A'){

                        ch=(char)(temp+add);

                        if(ch>'Z'){

                                ch=(char)(ch-'Z'+'A'-1);

                        }

                        key[i][0]=(char)temp;

                        key[i][1]=(char)(ch);

                        if(temp=='Z'){

                                temp=(char)('a'-1);

                        }

                }

                else if(temp<='z' && temp>='a'){

                        ch=(char)(temp+add);

                        if(ch>'z'){

                                ch=(char)(ch-'z'+'a'-1);

                        }

                        key[i][0]=(char)temp;
                        key[i][1]=(char)(ch);
                }
        }
}
public String doEncryption(String s){
        String cipherText="";
        for(int i=0;i<s.length();i++){
                for(int j=0;j<key.length;j++){
                        if(s.charAt(i)==key[j][0]){
                                cipherText+=key[j][1];
                        }
                }
        }
        return cipherText;
}
public void doDecryption(String s){
        String plainText="";
        for(int i=0;i<s.length();i++){
                for(int j=0;j<key.length;j++){
                        if(s.charAt(i)==key[j][1]){
                                plainText+=key[j][0];
                        }
                }
        }
        System.out.println("Decrypted Text : " + plainText);
```

```
	}
	public static void main(String args[]){
		S_Box s=new S_Box();
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter Message : ");
		String plaintext=sc.nextLine();
		String encrypted = s.doEncryption(plaintext);
		System.out.println("Encrypted Text : " + encrypted);
		s.doDecryption(encrypted);
	}
}
```

OUTPUT:-

```
D:\sem_5\network_security>javac S_Box.java


D:\sem_5\network_security>java S_Box
Enter Message :
hello
Encrypted Text : ifmmp
Decrypted Text : hello
```

Question 8-DES

```
import javax.crypto.*;
import javax.crypto.spec.*;
import java.util.Scanner;

class DES{

        private SecretKey secretKey;

        DES() throws Exception{

                secretKey=KeyGenerator.getInstance("DES").generateKey();

        }

        private byte[] doEncryption(String plainText) throws Exception{



                Cipher cipher=Cipher.getInstance("DES");

                cipher.init(Cipher.ENCRYPT_MODE,secretKey);

                return cipher.doFinal(plainText.getBytes());

        }

        private byte[] doDecryption(String cipherText) throws Exception{

                Cipher cipher=Cipher.getInstance("DES");

                cipher.init(Cipher.DECRYPT_MODE,secretKey);

                return cipher.doFinal(cipherText.getBytes());

        }

        public static void main(String args[]) throws Exception{

                Scanner sc=new Scanner(System.in);

                System.out.println("Enter Message : ");

                String plainText=sc.nextLine();

                DES DES=new DES();

                String cipherText=new String(DES.doEncryption(plainText));

                System.out.println("Encrypted Text : " + cipherText);

                System.out.println("Encrypted Text : " + new String(DES.doDecryption(cipherText)));
```

```
	}
}
```

OUTPUT:-

```
--------------------------------------------------------------------------------------------------------------------------
D:\sem_5\network_security>java DES
Enter Message :
helloWOrld
Encrypted Text : ??qM?åa?òb‼Otiª÷
Encrypted Text : helloWOrld
--------------------------------------------------------------------------------------------------------------------------
```

Question 9:AES

```
import javax.crypto.*;
import javax.crypto.spec.*;
import java.util.Scanner;
class AES{
	private byte[] key;
	AES(){
		key="kHFlksfddsaKHBDS".getBytes();
	}
	private byte[] doEncryption(String plainText) throws Exception{
		SecretKeySpec secretKey=new SecretKeySpec(key,"AES");
		Cipher cipher=Cipher.getInstance("AES");
		cipher.init(Cipher.ENCRYPT_MODE,secretKey);
		return cipher.doFinal(plainText.getBytes());
	}
	private byte[] doDecryption(String cipherText) throws Exception{
		SecretKeySpec secretKey=new SecretKeySpec(key,"AES");
		Cipher cipher=Cipher.getInstance("AES");
		cipher.init(Cipher.DECRYPT_MODE,secretKey);
		return cipher.doFinal(cipherText.getBytes());
	}
	public static void main(String args[]) throws Exception{
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter Message : ");
		String plainText=sc.nextLine();
		AES aes=new AES();
		String cipherText=new String(aes.doEncryption(plainText));
		System.out.println("Encrypted Text : " + cipherText);
		System.out.println("Encrypted Text : " + new String(aes.doDecryption(cipherText)));
	}
}
```

OUTPUT:-

```
------------------------------------------------------------------------------------------------------------------------
D:\sem_5\network_security>java AES
Enter Message :
helloWOrld
Encrypted Text : [EÇw_♠?~??m??$<è
Encrypted Text : helloWOrld
------------------------------------------------------------------------------------------------------------------------
```

Question 10:RSA

```
import java.security.*;

import javax.crypto.*;

import javax.crypto.spec.*;

class RSAEncryption{

        public KeyPairGenerator keygenerator;

        public KeyPair myKey;

        Cipher c;

        public RSAEncryption() throws Exception{

                keygenerator = KeyPairGenerator.getInstance("RSA");

                keygenerator.initialize(1024) ;

                myKey = keygenerator.generateKeyPair();

                c = Cipher.getInstance("RSA");

        }

        public byte[] doEncryption(String s) throws Exception{

                c.init(Cipher.ENCRYPT_MODE,myKey.getPublic());

                byte[] text = s.getBytes();

                byte[] textEncrypted = c.doFinal(text);

                return(textEncrypted);

        }

        public String doDecryption(byte[] s)throws Exception{

                c.init(Cipher.DECRYPT_MODE,myKey.getPrivate());

                byte[] textDecrypted = c.doFinal(s);

                return(new String(textDecrypted));
```

```
	}
	public static void main(String[] argv) throws Exception{
		RSAEncryption d=new RSAEncryption();
		byte[] str=d.doEncryption("BipinRupadiya");
		System.out.println("Encrypted String : "+str);
		System.out.println("Encrypted String : "+d.doDecryption(str));
	}
}
```

OUTPUT:-

```
D:\sem_5\network_security>javac RSAEncryption.java


D:\sem_5\network_security>java RSAEncryption

Original Text  : Plain text which need to be encrypted by Java RSA Encryption in ECB Mode

Encrypted Text :
YuApmia7Gif/DCzNZNOPxyaE/Uu1Oh1Ljf9pQ5FXY8PW64ttY1Dy+LTnS3Y5ft3iafxMDUjdrYB6b2wTVJ+Tiu
STRIAEK+d5Dz71eWkLZQoaPp6/txAJlOp6VFgPgT3tI8flA7fgIUl9mbdRIzU5bay91QQwyFbGCDDIG10ODk6
3ykY8o2bRA3CATAPZ6LQJrnzpM25uJbfs24bv7qNrXjXs8VaY/f+xHFHzoE6a8ojHNrk3ZxL/xUwWL8cwj5V0l
Y2iB05rIGcwONy4zDu+AaoQTIGvNWsroplmeM0GsTEb456+emtv6g35KJuqAHY/ct4EgYE3Ej1tEboO5uafot
+fpGkbrZoH1HebIMq1gihqH3K/bWRFkJ19JqmmEJR95ZlZ8YqnpTDJbhWnF8O6FDz5uA6P/tZEpXLkJyKSjl8
xfMNYBCXGBMLDga/8xs0v/SjvdhpYHkilslXHpkECQ+7DOoDtijpwRVNrfnBckRfmWxuuxRvdJysW1VFgOdfx
7s40cDqmkryRLY8hwzhc/Zvc1XL35mVIhOacIHdU6Ei+oHtsjcBBosuzaSWx/yyfmQgThBJEVvvVobEAnsl+ND
H2tyMd2SBPxx6kItZNHIVMB+9tVjbYGgVau0MpavaCMcIEzhpQPait76n0UCraDyA22ZIoN9PfXB0+x/YShso
=

Exception in thread "main" java.security.NoSuchAlgorithmException: Cannot find any provider
supporting RSA/ECB/RSA/ECB/OAEPWITHSHA-512ANDMGF1PADDING
```

Question 11:-SHA

```
import java.util.Scanner;

import java.math.*;

import java.security.*;

class SHA{

        private String doEncryption(String text) throws Exception{

                MessageDigest md=MessageDigest.getInstance("SHA-1");

                byte[] msg=md.digest(text.getBytes());

                BigInteger bigInt=new BigInteger(1,msg);

                String hashValue=bigInt.toString(16);

                while(hashValue.length()<32)

                        hashValue+=0+hashValue;

                return hashValue;

        }

        public static void main(String args[]) throws Exception{

                SHA sha=new SHA();

                Scanner sc=new Scanner(System.in);

                System.out.println("Enter Message : ");

                String text=sc.nextLine();

                System.out.println("Hash Text : " + sha.doEncryption(text));

        }

}
```

OUTPUT:-

```
------------------------------------------------------------------------------------------------------------------------

D:\sem_5\network_security>java SHA

Enter Message :
```

```
helloWOrld

Hash Text : 450b1797debbcb16daa2b46dd64926b97f01eac6
```

Question 12:-MD5

```
import java.util.Scanner;

import java.math.*;

import java.security.*;

class MD5{

	private String doEncryption(String text) throws Exception{

		MessageDigest md=MessageDigest.getInstance("MD5");

		byte[] msg=md.digest(text.getBytes());

		BigInteger bigInt=new BigInteger(1,msg);

		String hashValue=bigInt.toString(16);

		while(hashValue.length()<32)

			hashValue+=0+hashValue;

		return hashValue;

	}

	public static void main(String args[]) throws Exception{

		MD5 MD5=new MD5();

		Scanner sc=new Scanner(System.in);

		System.out.println("Enter Message : ");

		String text=sc.nextLine();

		System.out.println("Hash Text : " + MD5.doEncryption(text));

	}
}
```

OUTPUT:-

```
-----------------------------------------------------------------------------------------------------------------------

D:\sem_5\network_security>java MD5

Enter Message :

helloWOrld

Hash Text : a8849245f11e8297678fe957a6869ddf

-----------------------------------------------------------------------------------------------------------------------
```

Question 13:- Implement authentication Service. The sender sends password in encrypted format to the receiver, the receiver checks

the password (after decrypting) in its database/array and replies back as success or failure.(Keys are already shared)

Prg1_client.java:-

```
import java.io.*;

import java.util.*;

import java.net.*;

class prg1_client

{

	public static void main(String args[])throws Exception

	{

		Scanner sc=new Scanner(System.in);

		System.out.print("Enter the username");

		String uname=sc.nextLine();

		System.out.print("Enter the password");
		String pwd=sc.nextLine();

		Socket s=new Socket("127.0.0.1",5678);

		DataOutputStream dos=new DataOutputStream(s.getOutputStream());

		dos.writeUTF(uname);

		conversion c=new conversion();

		String encrypt_pwd=c.convert((3*-1),pwd);

		dos.writeUTF(encrypt_pwd);

		DataInputStream dis=new DataInputStream(s.getInputStream());

		String msg=dis.readUTF();

		System.out.println(msg);

		s.close();

	}

}

class conversion

{

	char temp,t;

	int asci=0;

	public String convert(int key,String inputmsg)

	{

		String op="";

		int min=0,max=0,flag=0;

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

		{

			temp=inputmsg.charAt(i);

			asci=temp+key;
```

```
if (temp>=97&&temp<=122)
{

        min=97;
        max=122;
        flag=1;
}
else if(temp>=48&&temp<=57)
{
        min=48;
        max=57;
        flag=1;
}
else if(temp>=65&&temp<=90)
{
        min=65;
        max=90;
        flag=1;
}
else
        flag=0;

if(flag==1)
{
        if(asci>max)
```

```
                    {
                        int rem=asci-max;
                        t=(char)((min-1)+rem);
                    }
                    else if(asci<min)
                    {
                        int rem=min-asci;
                        t=(char)((max+1)-rem);
                    }
                    else
                    {
                        t=(char)(asci);
                    }
                    op=op+t;
                }
                else
                {
                    op=op+temp;
                }
            }
            return op;
        }
}
```

Prg1_server.java:-

```
import java.util.*;

import java.io.*;

import java.net.*;

import java.sql.*;

class prg1_server

{


        public static void main(String args[])throws Exception

        {

                ServerSocket ss=new ServerSocket(5678);

                Socket s=ss.accept();

                DataInputStream dis=new DataInputStream(s.getInputStream());

                String uname=dis.readUTF();

                String e_password=dis.readUTF();

                System.out.println(uname);

                System.out.println(e_password);

                conversion c=new conversion();

                String d_password=c.convert(3,e_password);


                Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/user","root","");


                Statement st= con.createStatement();


                String sql="select password from user_details where user_id='"+uname+"';";
```

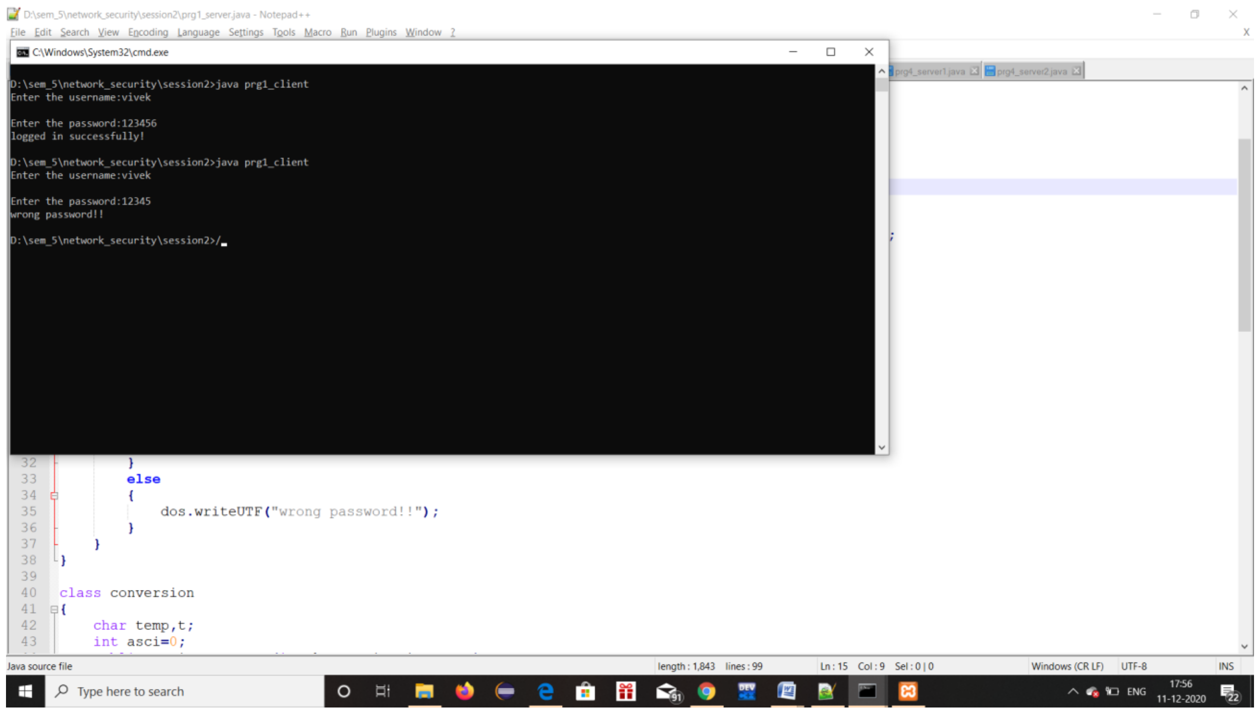
```
				ResultSet rs=st.executeQuery(sql);
				rs.next();
				String password=rs.getString(1);
				con.close();
				DataOutputStream dos=new DataOutputStream(s.getOutputStream());
				if(password.equals(d_password))
				{
					dos.writeUTF("logged in successfully!");
				}
				else
				{
					dos.writeUTF("wrong password!!");
				}
		}
}

class conversion
{
		char temp,t;
		int asci=0;
		public String convert(int key,String inputmsg)
		{
				String op="";
				int min=0,max=0,flag=0;
				for(int i=0;i<inputmsg.length();i++)
```

```
{
        temp=inputmsg.charAt(i);
        asci=temp+key;
        if (temp>=97&&temp<=122)
        {

                min=97;
                max=122;
                flag=1;
        }
        else if(temp>=48&&temp<=57)
        {
                min=48;
                max=57;
                flag=1;
        }
        else if(temp>=65&&temp<=90)
        {
                min=65;
                max=90;
                flag=1;
        }
        else
                flag=0;
```

```
    }

            if(flag==1)

            {

                if(asci>max)

                {

                    int rem=asci-max;

                    t=(char)((min-1)+rem);

                }

                else if(asci<min)

                {

                    int rem=min-asci;

                    t=(char)((max+1)-rem);

                }

                else

                {

                    t=(char)(asci);

                }

                op=op+t;

            }

            else

            {

                op=op+temp;

            }

        }

        return op;

    }
```

}

OUTPUT:-


D:\sem_5\network_security\session2\prg1_server.java - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
C:\Windows\System32\cmd.exe
prg4_server1.java
prg4_server2.java
Java source file
length : 1,843 lines : 99
Ln : 15 Col : 9 Sel : 0 | 0
Windows (CR LF) UTF-8 INS
Type here to search
ENG
17:56
11-12-2020

```
D:\sem_5\network_security\session2>java prg1_client
Enter the username:vivek

Enter the password:123456
logged in successfully!

D:\sem_5\network_security\session2>java prg1_client
Enter the username:vivek

Enter the password:12345
wrong password!!

D:\sem_5\network_security\session2>/_
```

```
            }
            else
            {
                dos.writeUTF("wrong password!!");
            }
        }
}

class conversion
{
    char temp,t;
    int asci=0;
```

Question 14: Implement authentication Service. The sender sends password in encrypted format to the receiver,

the receiver checks the password (after decrypting and applying hash) in its database/array and replies

back as success or failure. (Note: Here the password is stored as hash in database).

Prg2_client.java:-

import java.io.*;

import java.util.*;

```
import java.net.*;

class prg2_client

{

        public static void main(String args[])throws Exception

        {

                Scanner sc=new Scanner(System.in);

                System.out.print("Enter the username");

                String uname=sc.nextLine();

                System.out.print("Enter the password");

                String pwd=sc.nextLine();

                Socket s=new Socket("127.0.0.1",5678);

                DataOutputStream dos=new DataOutputStream(s.getOutputStream());

                dos.writeUTF(uname);

                conversion c=new conversion();

                String encrypt_pwd=c.convert((3*-1),pwd);

                dos.writeUTF(encrypt_pwd);

                DataInputStream dis=new DataInputStream(s.getInputStream());

                String msg=dis.readUTF();

                System.out.println(msg);

                s.close();

        }

}

class conversion

{

        char temp,t;
```

```
int asci=0;

public String convert(int key,String inputmsg)

{

        String op="";

        int min=0,max=0,flag=0;

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

        {

                temp=inputmsg.charAt(i);

                asci=temp+key;

                if (temp>=97&&temp<=122)

                {


                        min=97;

                        max=122;

                        flag=1;

                }

                else if(temp>=48&&temp<=57)

                {

                        min=48;

                        max=57;

                        flag=1;

                }

                else if(temp>=65&&temp<=90)

                {

                        min=65;
                        max=90;
                        flag=1;
                }
                else
                        flag=0;

                if(flag==1)
                {
                        if(asci>max)
                        {
                                int rem=asci-max;
                                t=(char)((min-1)+rem);
                        }
                        else if(asci<min)
                        {
                                int rem=min-asci;
                                t=(char)((max+1)-rem);
                        }
                        else
                        {
                                t=(char)(asci);
                        }
                        op=op+t;
                }
                else
```

```
                        {
                                op=op+temp;
                        }
                }
                return op;
        }
}
```

Prg2_server.java:-

```
import java.util.*;
import java.io.*;
import java.net.*;
import java.math.*;
import java.security.*;
import java.sql.*;
class prg2_server
{

        public static void main(String args[])throws Exception
        {
                ServerSocket ss=new ServerSocket(5678);
                Socket s=ss.accept();
                DataInputStream dis=new DataInputStream(s.getInputStream());
                String uname=dis.readUTF();
                String e_password=dis.readUTF();
                conversion c=new conversion();
        }

                String d_password=c.convert(3,e_password);

                SHA hash=new SHA();
                String hash_password=hash.doEncryption(d_password);

                Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/user","root","");

                Statement st= con.createStatement();

                String sql="select hash_code from user_details where user_id='"+uname+"';";

                ResultSet rs=st.executeQuery(sql);
                rs.next();
                String db_hash_code=rs.getString(1);
                con.close();
                DataOutputStream dos=new DataOutputStream(s.getOutputStream());
                if(hash_password.equals(db_hash_code))
                {
                        dos.writeUTF("logged in successfully!");
                }
                else
                {
                        dos.writeUTF("wrong password!!");
                }
        }
```

```
}
class conversion
{
        char temp,t;
        int asci=0;
        public String convert(int key,String inputmsg)
        {
                String op="";
                int min=0,max=0,flag=0;
                for(int i=0;i<inputmsg.length();i++)
                {
                        temp=inputmsg.charAt(i);
                        asci=temp+key;
                        if (temp>=97&&temp<=122)
                        {

                                min=97;
                                max=122;
                                flag=1;
                        }
                        else if(temp>=48&&temp<=57)
                        {
                                min=48;
                                max=57;
                                flag=1;
```

```
}
else if(temp>=65&&temp<=90)
{
	min=65;
	max=90;
	flag=1;
}
else
	flag=0;

if(flag==1)
{
	if(asci>max)
	{
		int rem=asci-max;
		t=(char)((min-1)+rem);
	}
	else if(asci<min)
	{
		int rem=min-asci;
		t=(char)((max+1)-rem);
	}
	else
	{
		t=(char)(asci);
```

```
                        }
                        op=op+t;
                    }
                    else
                    {
                        op=op+temp;
                    }
                }
                return op;
        }
}

class SHA{
        public String doEncryption(String text) throws Exception{
                MessageDigest md=MessageDigest.getInstance("SHA-1");
                byte[] msg=md.digest(text.getBytes());
                BigInteger bigInt=new BigInteger(1,msg);
                String hashValue=bigInt.toString(16);
                while(hashValue.length()<32)
                        hashValue+=0+hashValue;
                return hashValue;
        }
}
```

OUTPUT:-

```
D:\sem_5\network_security\session2>javac prg2_client.java


D:\sem_5\network_security\session2>java prg2_client

Enter the username:vivek

Enter the password:123456

logged in successfully!
```

Question 15:- Implement a firewall that behaves like forwarder. It does not forward the packet that contains the word "terrorist".

Prg3_client.java

```
import java.io.*;

import java.util.*;

import java.net.*;

class prg3_client

{

        public static void main(String args[])throws Exception

        {

                Scanner sc=new Scanner(System.in);

                System.out.print("Enter packet you want to send to server:");

                String packet=sc.nextLine();

                Socket s=new Socket("127.0.0.1",1234);

                DataOutputStream dos=new DataOutputStream(s.getOutputStream());

                dos.writeUTF(packet);

                DataInputStream dis=new DataInputStream(s.getInputStream());
```

```
		String server_msg=dis.readUTF();
		System.out.println(server_msg);
		s.close();

	}
}
```

Prg3_firewall.java:-

```
import java.io.*;
import java.util.*;
import java.net.*;
class prg3_firewall
{
	public static void main(String args[])throws Exception
	{
		ServerSocket ss=new ServerSocket(1234);
		Socket s1=ss.accept();
		DataInputStream dis=new DataInputStream(s1.getInputStream());
		String client_msg=dis.readUTF();
		String chk_pck=client_msg.toLowerCase();
		String threat="terrorist";
		StringTokenizer st=new StringTokenizer(chk_pck," ");
		String err="";
		int flag=0;
		DataOutputStream dos=new DataOutputStream(s1.getOutputStream());
```

```
while(st.hasMoreTokens())
{
        if(threat.equals(st.nextToken()))
        {

                err="Threat in package.. can't be delivered ";
                dos.writeUTF(err);
                s1.close();
                flag=1;
                break;
        }
}
if(flag==0)
{
        Socket s2=new Socket("127.0.0.1",5678);
        DataOutputStream dos1=new DataOutputStream(s2.getOutputStream());
        dos1.writeUTF(client_msg);
        DataInputStream dis1=new DataInputStream(s2.getInputStream());
        String ack=dis1.readUTF();
        if(ack.equals("1"))
        {
                dos.writeUTF("packet recieved");
        }
        else
        {
```

```
                                        dos.writeUTF("unable to reach server!!");
                              }
                              s1.close();
                              s2.close();
                    }
                    ss.close();
          }
}
```

Prg3_server.java:-

```
import java.io.*;
import java.util.*;
import java.net.*;
class prg3_server
{
          public static void main(String args[])throws Exception
          {
                    ServerSocket ss=new ServerSocket(5678);
                    Socket s=ss.accept();
                    DataInputStream dis=new DataInputStream(s.getInputStream());
                    String client_msg=dis.readUTF();
                    System.out.println("client packet:"+client_msg);
                    DataOutputStream dos=new DataOutputStream(s.getOutputStream());
                    dos.writeUTF("1");
                    s.close();
```

```
			ss.close();

		}

}
```

OUTPUT:-

```
D:\sem_5\network_security\session2>java prg3_client

Enter packet you want to send to server:vivek

packet recieved


D:\sem_5\network_security\session2>java prg3_firewall


D:\sem_5\network_security\session2>java prg3_server

client packet:vivek

D:\sem_5\network_security\session2>
```

Question 16:- Implement NAT functionality. The NAT works like forwarder, that will forward to appropriate receiver.

Prg4_client.java:-

```
import java.io.*;

import java.util.*;

import java.net.*;

class prg4_client

{
	{

		public static void main(String args[])throws Exception

		{

				Scanner sc=new Scanner(System.in);

				String numbers="";

				System.out.print("Enter set of numbers you want to send :");

				String num=sc.nextLine();

				Socket s=new Socket("127.0.0.1",1234);

				DataOutputStream dos=new DataOutputStream(s.getOutputStream());

				dos.writeUTF(num);

				DataInputStream dis=new DataInputStream(s.getInputStream());

				String server_msg=dis.readUTF();

				System.out.println(server_msg);

				s.close();


		}

}
```

Prg4_forwarder.java:-

```
import java.io.*;

import java.util.*;

import java.net.*;

class prg4_forwarder

{

		public static void main(String args[])throws Exception

		{
```

```
ServerSocket ss=new ServerSocket(1234);

Socket s=ss.accept();

DataInputStream dis=new DataInputStream(s.getInputStream());

DataOutputStream dos=new DataOutputStream(s.getOutputStream());

String client_msg=dis.readUTF();

String n=client_msg.toLowerCase();

StringTokenizer st=new StringTokenizer(n," ");

int flag1=0,flag2=0,count=0;

Socket s1=new Socket("127.0.0.1",5678);

Socket s2=new Socket("127.0.0.1",5679);

DataOutputStream dos1=new DataOutputStream(s1.getOutputStream());

DataInputStream dis1=new DataInputStream(s1.getInputStream());

DataOutputStream dos2=new DataOutputStream(s2.getOutputStream());

DataInputStream dis2=new DataInputStream(s2.getInputStream());

String server1_msg="",server2_msg="";

while(st.hasMoreTokens())

{

        int num=Integer.parseInt(st.nextToken());

        if(num%2==0)

        {

                server2_msg=server2_msg+" "+num;

        }

        else

        {

                server1_msg=server1_msg+" "+num;
                }
            }
            dos1.writeUTF(server1_msg);
            dos2.writeUTF(server2_msg);
            String ack1=dis1.readUTF();
            String ack2=dis2.readUTF();
            if(ack1.equals("1")&&ack2.equals("1"))
            {
                dos.writeUTF("packets delivered to servers");
            }
            else
            {
                dos.writeUTF("packets not delivered to servers");
            }
            ss.close();
            s.close();
            s1.close();
            s2.close();
        }
}
```

Prg4_server1.java:-

```
import java.io.*;
import java.util.*;
import java.net.*;
class prg4_server1
```

```
{
	public static void main(String args[])throws Exception
	{
		ServerSocket ss=new ServerSocket(5678);
		Socket s=ss.accept();
		DataInputStream dis=new DataInputStream(s.getInputStream());
				DataOutputStream dos=new DataOutputStream(s.getOutputStream());
		String client_msg=dis.readUTF();
		if(client_msg.equals(""))
		{
			dos.writeUTF("0");
		}
		else
		{
		System.out.println("client packet:"+client_msg);

		dos.writeUTF("1");
		}
		s.close();
		ss.close();
	}
}
```

Prg4_server2.java:-

```
import java.io.*;
import java.util.*;
import java.net.*;

class prg4_server2

{

	public static void main(String args[])throws Exception

	{

		ServerSocket ss=new ServerSocket(5679);

		Socket s=ss.accept();

		DataInputStream dis=new DataInputStream(s.getInputStream());

		DataOutputStream dos=new DataOutputStream(s.getOutputStream());

		String client_msg=dis.readUTF();

		if(client_msg.equals(""))

		{

			dos.writeUTF("0");

		}

		else

		{

		System.out.println("client packet:"+client_msg);


		dos.writeUTF("1");

		}

		s.close();

		ss.close();

	}

}
```

OUTPUT:-

```
D:\sem_5\network_security\session2>java prg4_client

Enter set of numbers you want to send :1 2 3 4 5 6 7 8 9

packets delivered to servers


D:\sem_5\network_security\session2>java prg4_forwarder


D:\sem_5\network_security\session2>java prg4_server2

client packet: 2 4 6 8


D:\sem_5\network_security\session2>java prg4_server1

client packet: 1 3 5 7 9
```

Question 17:- Implement a program to demonstrate the functioning of a KDC. There are three entities: sender, receiver and KDC. Assume that Sender and Receiver have already established their own individual permanent secret keys with KDC. The sender requests the KDC to issue a session key to communicate with receiver. The KDC is supposed to give session key information to sender in a secure way. The same session key is also to be communicated to the receiver securely. Use a suitable protocol to achieve the above functionality.

Client.java:-

```java
import java.io.DataInputStream;

import java.net.*;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

class Client{

        static String receiverid;

        static SecretKeySpec receiverkey;

        public static void main(String args[]) throws Exception{

                System.out.println("client");

    receiverid="receiver123";

    receiverkey=new SecretKeySpec("12345678".getBytes(),"DES");


    Socket s=new Socket("localhost",9090);

        DataInputStream dis=new DataInputStream(s.getInputStream());


        byte[] encryptedsenderid=new byte[dis.readInt()];

        dis.readFully(encryptedsenderid);


        byte[] encryptedreceiverid=new byte[dis.readInt()];

        dis.readFully(encryptedreceiverid);


        byte[] encryptedsessionkeyclient=new byte[dis.readInt()];

        dis.readFully(encryptedsessionkeyclient);


        Cipher cipher=Cipher.getInstance("DES");

        cipher.init(Cipher.DECRYPT_MODE,receiverkey);

        byte[] senderid=cipher.doFinal(encryptedsenderid);

        System.out.println("sender id" +new String(senderid));
        byte[] receiverid=cipher.doFinal(encryptedreceiverid);

        System.out.println("receiverid" +new String(receiverid));

        byte[] sessionkey=cipher.doFinal(encryptedsessionkeyclient);

        System.out.println("sessionkey" + new String(sessionkey));

        }

}
```

Kdc1.java:-

```
import java.io.DataOutputStream;

import java.net.ServerSocket;

import java.net.Socket;

import java.security.SecureRandom;

import javax.crypto.Cipher;

import javax.crypto.spec.SecretKeySpec;


class Kdc1{

        public static void main(String args[]) throws Exception{

                SecretKeySpec senderkey,receiverkey;

                byte [] sessionkey,encryptedsessionkey;

                String senderid,receiverid;

                        System.out.println("KDC");

                        receiverid="receiver123";

                        senderid="sender123";

                        receiverkey=new SecretKeySpec("12345678".getBytes(),"DES");

                        senderkey=new SecretKeySpec("87654321".getBytes(),"DES");

                        ServerSocket ss=new ServerSocket(8080);
```

```
                    Socket s=ss.accept();

                         sessionkey=generateSessionKey();

                         System.out.println("sessionkey" +new String(sessionkey));

                         DataOutputStream dos=new DataOutputStream(s.getOutputStream());

                         Cipher cipher=Cipher.getInstance("DES");

                              cipher.init(Cipher.ENCRYPT_MODE,senderkey);

                              encryptedsessionkey=cipher.doFinal(sessionkey);

                              cipher.init(Cipher.ENCRYPT_MODE,receiverkey);

                              byte[] encryptedreceiverid=cipher.doFinal(receiverid.getBytes());

                              byte[] encryptedsenderid=cipher.doFinal(senderid.getBytes());

                              byte[] encryptedsessionkeyclient=cipher.doFinal(sessionkey);


                        dos.writeInt(encryptedsessionkey.length);

                        dos.write(encryptedsessionkey,0,encryptedsessionkey.length);


                        dos.writeInt(encryptedsenderid.length);

                        dos.write(encryptedsenderid,0,encryptedsenderid.length);


                        dos.writeInt(encryptedreceiverid.length);

                        dos.write(encryptedreceiverid,0,encryptedreceiverid.length);


                        dos.writeInt(encryptedsessionkeyclient.length);

                        dos.write(encryptedsessionkeyclient,0,encryptedsessionkeyclient.length);

  }

public static byte [] generateSessionKey() throws Exception
```

```
            {
                        byte[] sessionkey=new byte[8];
                        SecureRandom random = new SecureRandom();
                        random.nextBytes(sessionkey);
                        return sessionkey;
            }
}
```

Server.java:-

```
import java.io.DataOutputStream;
import java.io.DataInputStream;
import java.net.ServerSocket;
import java.net.*;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
class Server{
             static String senderid;
            static SecretKeySpec senderkey;
            static byte[] encryptedreceiverid,encryptedsenderid,encryptedsessionkeyclient;
            public static void main(String args[]) throws Exception{
                        System.out.println("Server");
      senderid="sender123";
      senderkey=new SecretKeySpec("87654321".getBytes(),"DES");
      getSessionInfoServer();
      ServerSocket ss=new ServerSocket(9090);
            Socket s=ss.accept();
```

```
        DataOutputStream dos=new DataOutputStream(s.getOutputStream());
        dos.writeInt(encryptedsenderid.length);
        dos.write(encryptedsenderid,0,encryptedsenderid.length);

        dos.writeInt(encryptedreceiverid.length);
        dos.write(encryptedreceiverid,0,encryptedreceiverid.length);

        dos.writeInt(encryptedsessionkeyclient.length);
        dos.write(encryptedsessionkeyclient,0,encryptedsessionkeyclient.length);

}
public static void getSessionInfoServer() throws Exception
        {
                Socket s=new Socket(InetAddress.getLocalHost(),8080);
                DataInputStream dis=new DataInputStream(s.getInputStream());

                byte[] encryptedsessionkey=new byte[dis.readInt()];
         dis.readFully(encryptedsessionkey);

                encryptedsenderid=new byte[dis.readInt()];
         dis.readFully(encryptedsenderid);

                encryptedreceiverid=new byte[dis.readInt()];
                dis.readFully(encryptedreceiverid);
                encryptedsessionkeyclient=new byte[dis.readInt()];
            dis.readFully(encryptedsessionkeyclient);


                Cipher cipher=Cipher.getInstance("DES");
                cipher.init(Cipher.DECRYPT_MODE,senderkey);
                byte[] sessionkey=cipher.doFinal(encryptedsessionkey);
                System.out.println("serversessionkey" +new String(sessionkey));
            }
    }
```

OUTPUT:-

```
D:\sem_5\network_security\session2> javac Server.java
D:\sem_5\network_security\session2>java Server
Server
serversessionkeyA��|�^


D:\sem_5\network_security\session2> java Kdc1
KDC


D:\sem_5\network_security\session2> java Client
client
sender idsender123
receiveridreceiver123
```

sessionkeyA��|�^