Program 1:-Implement Casear Cipher

Code:-

```
import java.util.*;
class CasearCipher
{
	public static String encryption(String str)
	{
		String enc_text="";
		for(int i=0;i<str.length();i++)
		{
			char ch=str.charAt(i);
			if(Character.isUpperCase(ch))
			{
				int asc_ch=(((int)ch+3)-65)%26+65;
				enc_text=enc_text+(char)asc_ch;

			}
			else
			{
				int asc_ch=(((int)ch+3)-97)%26+97;
				enc_text=enc_text+(char)asc_ch;
			}
		}
		return  enc_text;
	}
		public static String decryption(String str)
	{
```

```
        String enc_text="";
        for(int i=0;i<str.length();i++)
        {
                char ch=str.charAt(i);
                if(Character.isUpperCase(ch))
                {
                        int asc_ch=(((int)ch-3)-65)%26+65;
                        enc_text=enc_text+(char)asc_ch;

                }
                else
                {
                        int asc_ch=(((int)ch-3)-97)%26+97;
                        enc_text=enc_text+(char)asc_ch;
                }
        }
        return  enc_text;
}
public static void main(String args[])
{
        Scanner s=new Scanner(System.in);
        System.out.print("Enter your plain text::-");
        String str=s.nextLine();
        String enc_text=encryption(str);
        System.out.println("Your encrypted plain text::-"+enc_text);
        String dec_text=decryption(enc_text);
        System.out.println("Your decrypted plain text::-"+dec_text);
```

```
        }
}
```

Output:-

```
E:\SUPAN\NS>javac CasearCipher.java
E:\SUPAN\NS>java CasearCipher
Enter your plain text::-supanshah
Your encrypted plain text::-vxsdqvkdk
Your decrypted plain text::-supanshah
E:\SUPAN\NS>
```

Program 2:-Implement Substitution Cipher

Code:-

```
import java.util.*;
class SubstitutionCipher
{
        public static String encryption(String str,int n)
        {
                String enc_text="";
                for(int i=0;i<str.length();i++)
                {
                        char ch=str.charAt(i);
                        if(Character.isUpperCase(ch))
                        {
                                int asc_ch=(((int)ch+n)-65)%26+65;
                                enc_text=enc_text+(char)asc_ch;

                        }
                        else
```

```
            {
                int asc_ch=(((int)ch+n)-97)%26+97;
                enc_text=enc_text+(char)asc_ch;
            }
        }
        return  enc_text;
}
        public static String decryption(String str,int n)
{
        String enc_text="";
        for(int i=0;i<str.length();i++)
        {
            char ch=str.charAt(i);
            if(Character.isUpperCase(ch))
            {
                int asc_ch=(((int)ch-n)-65)%26+65;
                enc_text=enc_text+(char)asc_ch;

            }
            else
            {
                int asc_ch=(((int)ch-n)-97)%26+97;
                enc_text=enc_text+(char)asc_ch;
            }
        }
        return  enc_text;
}
```

```
	public static void main(String args[])
	{
		Scanner s=new Scanner(System.in);
		System.out.print("\nEnter your plain text::-");
		String str=s.nextLine();
		System.out.print("Enter how many shifts u want::-");
		int n=s.nextInt();
		String enc_text=encryption(str,n);
		System.out.println("Your encrypted plain text::-"+enc_text);
		String dec_text=decryption(enc_text,n);
		System.out.println("Your decrypted plain text::-"+dec_text);
	}
}
```

Output:-

```
E:\SUPAN\NS>javac SubstitutionCipher.java
E:\SUPAN\NS>java SubstitutionCipher
Enter your plain text::-supan
Enter how many shifts u want::-3
Your encrypted plain text::-vxsdq
Your decrypted plain text::-supan
E:\SUPAN\NS>
```

Program 3:-Implement Transposition Cipher

Code:-

```
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:-

```
E:\SUPAN\NS>javac Transposition.java
E:\SUPAN\NS>java Transposition
enter key(of unique alphabets):
shyervg
enter string :supansh
Cipher text:
ahunssp
decipher text:
supansh
E:\SUPAN\NS>
```

Program 4:-Implement P-BOX

Code:-

```
import java.util.*;
class P_Box
```

```
{
        static String encryption(String msg)
        {
                byte arr[]=new byte[10];
                byte msg_arr[]=new byte[10];
                msg_arr=msg.getBytes();
                arr[0]=msg_arr[8];
                arr[1]=msg_arr[7];
                arr[2]=msg_arr[6];
                arr[3]=msg_arr[4];
                arr[4]=msg_arr[9];
                arr[5]=msg_arr[5];
                arr[6]=msg_arr[3];
                arr[7]=msg_arr[1];
                arr[8]=msg_arr[2];
                arr[9]=msg_arr[0];
                return(new String(arr));
        }
        static String decryption(String msg)
        {
                byte arr[]=new byte[10];
                byte msg_arr[]=new byte[10];

                msg_arr=msg.getBytes();
                arr[0]=msg_arr[9];
                arr[1]=msg_arr[7];
                arr[2]=msg_arr[8];
```

```
                arr[3]=msg_arr[6];
                arr[4]=msg_arr[3];
                arr[5]=msg_arr[5];
                arr[6]=msg_arr[2];
                arr[7]=msg_arr[1];
                arr[8]=msg_arr[0];
                arr[9]=msg_arr[4];




                return(new String(arr));

        }
        public static void main(String args[])
        {
                Scanner s=new Scanner(System.in);
                System.out.print("Enter your string::-");
                String msg=s.nextLine();
                String encryption_txt=encryption(msg);
                System.out.println("Your encrypted text::-"+encryption_txt);
                String decryption_txt=decryption(encryption_txt);
                System.out.println("Your decrypted text::-"+decryption_txt);
        }

}
```

Output:-

```
E:\SUPAN\NS>javac P_Box.java
```

```
E:\SUPAN\NS>java P_Box
Enter your string::-supanshahh
Your encrypted text::-hahnhsaups
Your decrypted text::-supanshahh
E:\SUPAN\NS>
```

Program 5:-Implement S-BOX

Code:-

```
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.print("Enter your message:- ");
                String plaintext=sc.nextLine();
                String encrypted = s.doEncryption(plaintext);
                System.out.println("Encrypted Text : " + encrypted);
                s.doDecryption(encrypted);
        }
}
```

Output:-

```
E:\SUPAN\NS>javac S_Box.java
E:\SUPAN\NS>java S_Box
Enter your message:- supanshah
Encrypted Text : vxsdqvkdk
Decrypted Text : supanshah
E:\SUPAN\NS>
```

Program 6:-Implement One Time Pad using XOR

Code:-

```
import java.util.*;
class OneTimePad1
```

```
{
	static String generate_key(String msg)
	{
		String key="";
		Random str_ascii=new Random();

		for(int i=0;i<msg.length();i++)
		{
			int n=str_ascii.nextInt(26);
			key=key+(char)(n+97);
		}
		return key;

	}
	static String encryption(String msg,String k)
	{
		String enc_text="";
		for(int i=0;i<msg.length();i++)
		{
			char enc_val=(char)(msg.charAt(i)^k.charAt(i));
			enc_text=enc_text+enc_val;
		}
		return enc_text;
	}
	static String decryption(String msg,String k)
	{
		String dec_text="";
```

```
        for(int i=0;i<msg.length();i++)
        {
            char dec_val=(char)(msg.charAt(i)^k.charAt(i));
            dec_text=dec_text+dec_val;
        }
        return dec_text;
    }
    public static void main(String[]args)
    {
        Scanner s=new Scanner(System.in);
        System.out.print("Enter your string::-");
        String message=s.nextLine();
        String key=generate_key(message);
        String encrypted_msg=encryption(message,key);
        System.out.println("Your encrypted message is ::"+encrypted_msg);
        String decrypted_msg=decryption(encrypted_msg,key);
        System.out.println("Your decrypted message is ::"+decrypted_msg);

    }
}
```

Output:-

```
E:\SUPAN\NS>javac OneTimePad1.java
E:\SUPAN\NS>java OneTimePad1
Enter your string::-supanshah
Your encrypted message is :: ☻→
¶↓↓♥
Your decrypted message is ::supanshah
```

Program 7:-Implement DES

Code:-

```
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.print("Enter your 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("Decrypted Text : " + new
String(DES.doDecryption(cipherText)));
        }
}
```

Output:-

```
E:\SUPAN\NS>javac DES.java
E:\SUPAN\NS>java DES
Enter your message : supanshah
Encrypted Text : u?l?↓♠Ä♠ô2?êP??ì
Decrypted Text : supanshah
```

Program 8:-Implement AES

Code:-

```
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.print("Enter your 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("Decrypted Text : " + new
String(aes.doDecryption(cipherText)));
	}
}
```

Output:-

```
E:\SUPAN\NS>javac AES.java
E:\SUPAN\NS>java AES
Enter your message : supanshah
Encrypted Text : ûì▲???&L☼±÷?W6?%
Decrypted Text : supanshah
E:\SUPAN\NS>
```

Program 9:-Implement SHA

Code:-

```
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:-

```
E:\SUPAN\NS>javac SHA.java
E:\SUPAN\NS>java SHA
Enter Message :
supan
Hash Text : 5a46585abba26b3d5ea0a99f1fe4a2ed3c3864be
E:\SUPAN\NS>
```

Program 10:-Implement MD5

Code:-

```
import java.util.Scanner;
import java.math.*;
import java.security.*;
class MD5
{
	static String encryption(String message)throws Exception
	{
		MessageDigest md=MessageDigest.getInstance("MD5");
		byte[] msg=md.digest(message.getBytes());
		BigInteger b=new BigInteger(1,msg);
		String hashval=b.toString(16);
		while(hashval.length()<32)
		{
			hashval+=0+hashval;
		}
		return hashval;
	}
	public static void main(String args[])throws Exception
	{
		Scanner s=new Scanner(System.in);
		System.out.print("\nEnter message::-");
		String msg=s.nextLine();
		String enc=encryption(msg);
		System.out.println("Encrypted message::-"+enc);
```

```
        }
}
```

Output:-

```
E:\SUPAN\NS>javac MD5.java
E:\SUPAN\NS>java MD5
Enter message::-supanshah
Encrypted message::-f5ee99632ac3201827f565fc96398854
E:\SUPAN\NS>
```

Program 11:-Implement RSA

Code:-

```
import java.io.DataInputStream;
import java.io.IOException;
import java.math.BigInteger;
import java.util.Random;
public class RSADemo{
    private BigInteger P;
    private BigInteger Q;
    private BigInteger N;
    private BigInteger PHI;
    private BigInteger e;
    private BigInteger d;
    private int maxLength = 1024;
    private Random R;
    public RSADemo(){
        R = new Random();
        P = BigInteger.probablePrime(maxLength, R);
         Q = BigInteger.probablePrime(maxLength, R);
```

```
    N = P.multiply(Q);
   PHI = P.subtract(BigInteger.ONE).multiply(  Q.subtract(BigInteger.ONE));
    e = BigInteger.probablePrime(maxLength / 2, R);
    while (PHI.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(PHI) < 0)
    {
        e.add(BigInteger.ONE);
    }
    d = e.modInverse(PHI);
}

public RSADemo(BigInteger e, BigInteger d, BigInteger N)
{
    this.e = e;
    this.d = d;
    this.N = N;
}

public static void main (String [] arguments) throws IOException
{
    RSADemo rsa = new RSADemo();
    DataInputStream input = new DataInputStream(System.in);
    String inputString;
    System.out.println("Enter message you wish to send.");
    inputString = input.readLine();
    System.out.println("Encrypting the message: " + inputString);
    System.out.println("The message in bytes is:: "
            + bToS(inputString.getBytes()));
```

```
    // encryption
    byte[] cipher = rsa.encryptMessage(inputString.getBytes());
    // decryption
    byte[] plain = rsa.decryptMessage(cipher);
    System.out.println("Decrypting Bytes: " + bToS(plain));
    System.out.println("Plain message is: " + new String(plain));
}

private static String bToS(byte[] cipher)
{
    String temp = "";
    for (byte b : cipher)
    {
        temp += Byte.toString(b);
    }
    return temp;
}

// Encrypting the message
public byte[] encryptMessage(byte[] message)
{
    return (new BigInteger(message)).modPow(e, N).toByteArray();
}

// Decrypting the message
public byte[] decryptMessage(byte[] message)
{
```

```
        return (new BigInteger(message)).modPow(d, N).toByteArray();
    }
}
```

Output:-

```
E:\SUPAN\NS>javac RSADemo.java
Note: RSADemo.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
E:\SUPAN\NS>java RSADemo
Enter message you wish to send.
hellosupan
Encrypting the message: hellosupan
The message in bytes is:: 104101108108111115117112971l0
Decrypting Bytes: 104101108108111115117112971l0
Plain message is: hellosupan
E:\SUPAN\NS>
```

Program 12:-Implement RSA

Code:-

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
class RSAEncryption
{
 public KeyPairGenerator keygenerator;
 public KeyPair myKey;
 Cipher c;
 public RSAEncryption() throws Exception
 {
```

```
 // Genrate the Key
 keygenerator = KeyPairGenerator.getInstance("RSA");
 keygenerator.initialize(1024) ;
 myKey = keygenerator.generateKeyPair();



 // Create the cipher
 c = Cipher.getInstance("RSA");



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

    // Initialize the cipher for encryption
    c.init(Cipher.ENCRYPT_MODE,myKey.getPublic());

  //sensitive information
    byte[] text = s.getBytes();

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

  return(textEncrypted);
}
public String doDecryption(byte[] s)throws Exception
```

```
{

    // Initialize the same cipher for decryption
  c.init(Cipher.DECRYPT_MODE,myKey.getPrivate());

    // Decrypt the text
    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("SupanShah");
  System.out.println("Encrypted String : "+str);
  System.out.println("Decrypted String : "+d.doDecryption(str));

 }
}
```

Output:-

```
E:\SUPAN\NS>javac RSAEncryption.java
E:\SUPAN\NS>java RSAEncryption
Encrypted String : [B@1ed40e0
Decrypted String : SupanShah
E:\SUPAN\NS>
```

Program 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.

Code:-

Sender Side:-

```
import java.io.*;
import java.net.*;
import java.util.*;
class SenderSide
{
	static String serverside_msg(byte[] arr)
	{

		String dec_txt=" ";
		int i=0;
		while(arr[i]!=0)
		{
			dec_txt= dec_txt+(char)arr[i];
			i++;
		}

		return dec_txt;
	}
	static void generate_key(String msg)
	{
		String key="";
		Random str_ascii=new Random();
```

```
                for(int i=0;i<msg.length();i++)
                {
                        int n=str_ascii.nextInt(26);
                        key=key+(char)(n+97);
                }
                try
                {
    FileWriter fw=new FileWriter("programkey.txt");
    fw.write(key);
    fw.close();
  }
                catch(Exception e)
                {
                         System.out.println(e);
                }
  System.out.println("Successfully written...");
}

     static String read_data() throws IOException
     {
                Scanner s=new Scanner(System.in);
                String filename=s.nextLine();
                int ch;
                String str="";
                FileInputStream in=new FileInputStream(filename +".txt");
                while((ch=in.read())!=-1)
                {
```

```
                str=str+(char)ch;
        }
        in.close();
        return str;
}
static String encryption(String msg)throws IOException
{
        System.out.print("\nEnter the file name for fetching key for encryption::-");
        String k=read_data();
        String enc_text="";
        for(int i=0;i<msg.length();i++)
        {
                char enc_val=(char)(msg.charAt(i)^k.charAt(i));
                enc_text=enc_text+enc_val;
        }
        return enc_text;
}
public static void main(String args[])throws Exception
{

        Scanner s=new Scanner(System.in);
        System.out.print("Enter the password::-");
        String str=s.nextLine();
        generate_key(str);
        String enc_text=encryption(str);

        DatagramSocket data_socket=new DatagramSocket(6789);
```

```
                InetAddress ip=InetAddress.getLocalHost();
                byte s_arr[]=null;
                s_arr=enc_text.getBytes();
                DatagramPacket data_packet=new
DatagramPacket(s_arr,s_arr.length,ip,1234);
                data_socket.send(data_packet);
                byte[] r_arr=new byte[65335];
                data_packet=new DatagramPacket(r_arr,r_arr.length);
                data_socket.receive(data_packet);
                System.out.println("\nServer Side message::-"+serverside_msg(r_arr));

        }
}
```

Receiver Side:-

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.sql.*;
class ReceiverSide
{

        static String read_data() throws IOException
        {
                Scanner s=new Scanner(System.in);
                String filename=s.nextLine();
                int ch;
                String str="";
```

```
        FileInputStream in=new FileInputStream(filename +".txt");
        while((ch=in.read())!=-1)
        {
                str=str+(char)ch;
        }
        in.close();
        return str;
}
static String client_msg(byte[] arr)
{
        String enc_txt="";
        if(arr==null)
                return null;
        int i=0;
        while(arr[i]!=0)
        {
                enc_txt=enc_txt+(char)arr[i];
                i++;
        }
        return enc_txt;
}
static String passwordCheck(String password)
{
        String msg="";
        String[] arr={"supan","nem","ashish","aqueed","parth"};
        for(int i=0;i<arr.length;i++)
        {
```

```
                String str=arr[i];


                if(password.equals(str))
                {
                        msg="Successfull!!!!";
                        break;
                }
                else
                {
                        msg="Unsuccessfull!!!";
                }
        }
        return msg;
}
static String decryption(String msg)throws IOException
{
        System.out.print("\nEnter the file name for fetching key for decryption::-");
        String k=read_data();
        String dec_text="";
        for(int i=0;i<msg.length();i++)
        {
                char dec_val=(char)(msg.charAt(i)^k.charAt(i));
                dec_text=dec_text+dec_val;
        }
        return dec_text;
}
```

```
	public static void main(String args[])throws Exception
	{
		Scanner s=new Scanner(System.in);
		DatagramSocket data_socket= new DatagramSocket(1234);
		byte[] rec_data=new byte[65335];
		DatagramPacket data_packet=new DatagramPacket(rec_data,rec_data.length);
		data_socket.receive(data_packet);
		String enc_txt=client_msg(rec_data);
		System.out.println("\n Client Side Message::-"+enc_txt);
		String dec_text=decryption(enc_txt);
		String msg_send_to_client=passwordCheck(dec_text);
		byte[] s_arr=new byte[65335];
		InetAddress ip=data_packet.getAddress();
		s_arr=msg_send_to_client.getBytes();
		data_packet=new DatagramPacket(s_arr,s_arr.length,ip,6789);
		data_socket.send(data_packet);
		data_socket.close();
	}
}
```

Output:-

```
E:\SUPAN\NS>javac SenderSide.java
E:\SUPAN\NS>java SenderSide
Enter the password::-supan
Successfully written...
Enter the file name for fetching key for encryption::-programkey
Server Side message::- Successfull!!!
E:\SUPAN\NS>
```

```
E:\SUPAN\NS>javac ReceiverSide.java

E:\SUPAN\NS>java ReceiverSide

 Client Side Message::-▲♣‼♀

Enter the file name for fetching key for decryption::-programkey

E:\SUPAN\NS>
```

Program 14:- Implement a firewall that behaves like forwarder. It does not forward the packet that contains the word "terrorist".

Code:-

Client Side:-

```
import java.util.*;

import java.io.*;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.net.SocketException;


public class Ass2Prog3ClientSide

{

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

        {

                Scanner sc=new Scanner(System.in);

                DatagramSocket ds=new DatagramSocket();

                InetAddress ip=InetAddress.getLocalHost();

                byte buf[]=null;

                String s;

                System.out.print("Enter your message: ");

                s=sc.nextLine();
```

```
                buf=s.getBytes();
                DatagramPacket dp=new DatagramPacket(buf,buf.length,ip,1233);
                ds.send(dp);

        }
}
```

Firewall:-

```
import java.util.*;
import java.io.*;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;

public class Ass2Prog3Firewall
{
        public static void main(String args[]) throws IOException
        {
                DatagramSocket ds=new DatagramSocket(1233);
                DatagramSocket ds1=new DatagramSocket();
                InetAddress ip=InetAddress.getLocalHost();
                byte buf[]=new byte[65535];
                byte buf1[]=new byte[65535];
                String s;
                DatagramPacket dp=new DatagramPacket(buf,buf.length);
                ds.receive(dp);
```

```
        s=new String(buf).trim();
        System.out.println("String successfully received by Firewall: "+s);
        if(checkString(s)==null)
        {
                buf1=s.getBytes();
                DatagramPacket dp1=new DatagramPacket(buf1,buf1.length,ip,1234);
                ds1.send(dp1);
        }
        else
        {
                System.out.println("Message "+s+" cannot be send!");
                checkString(s);
        }
}

public static String checkString(String s)throws IOException
{
        BufferedReader br=new BufferedReader(new FileReader("keywords.txt"));
        String temp="";
        while((temp=br.readLine())!=null)
        {
                if((s.toUpperCase()).contains(temp))
                {
                        return temp;
                }
        }
        return null;
```

```
        }
}
```

ServerSide:-

```
import java.util.*;
import java.io.*;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;

public class Ass2Prog3ServerSide
{
        public static void main(String args[]) throws IOException
        {
                DatagramSocket ds=new DatagramSocket(1234);
                InetAddress ip=InetAddress.getLocalHost();
                byte buf[]=new byte[65535];
                String es;
                DatagramPacket dp=new DatagramPacket(buf,buf.length);
                ds.receive(dp);
                es=new String(buf).trim();
                System.out.println("String successfully received: "+es);
        }
}
```

Output:-

E:\SUPAN\NS>javac Ass2Prog3ClientSide.java

E:\SUPAN\NS>java Ass2Prog3ClientSide

Enter your message: terroist

E:\SUPAN\NS>

E:\SUPAN\NS>javac Ass2Prog3Firewall.java

E:\SUPAN\NS>java Ass2Prog3Firewall

String successfully received by Firewall: terroist

Message terroist cannot be send!

Program 15:- Implement NAT functionality. The NAT works like forwarder, that will forward to appropriate receiver.

Code:-

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();
	}
}
```

Program 16:- 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.

Code:-

```
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;

public class prg5_KDC {
    static SecretKeySpec senderkey,receiverkey;
      static  byte [] sessionkey,encryptedsessionkey;
       static  String senderid,receiverid;
    public static void main(String[] args) throws Exception {
        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.l
ength);


    }
```

```
    public static byte [] generateSessionKey() throws Exception
      {
            sessionkey=new byte[8];
            SecureRandom random = new SecureRandom();
            random.nextBytes(sessionkey);
            return sessionkey;
      }
}
```

prg5_server.java

```
package dissertation;
import java.io.DataOutputStream;
import java.io.DataInputStream;
import java.net.ServerSocket;
import java.net.Socket;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

public class prg5_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.l
ength);

    }
    public static void getSessionInfoServer() throws Exception
      {
            Socket s=new Socket("localhost",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));
      }
}
```