Department of Computer Science

Gujarat University

Certificate

Roll No: 10 *Seat No:* _____

This is to certify that Mr./Ms. Gaurang Sanjaybhai Jani *student of MCA Semester – V has duly completed his/her term work for the semester ending in December 2020, in the subject of* Network Security *towards partial fulfillment of his/her Degree of Masters in Computer Applications.*

11/12/2020
Date of Submission

Internal Faculty

Head of Department

Department Of Computer Science
Rollwala Computer Centre
Gujarat University

MCA – 5

Subject: - Network Security

Name: - Gaurang Sanjaybhai Jani

Roll No.: - 10 **Exam Seat No.: -**

Sr. No.	Contents	Pg. No	Date	Signature
1	Assignment - 1	1	12/08/'20	
2	Assignment - 2	3	15/09/'20	
3	Assignment - 3	6	10/11/'20	
4	Assignment - 4	7	24/11/'20	

• Network Security •

Page No. 1.
Date

• Assignment - 1.

[1] List all Symmetric Key Algorithms.

→ ① AES → Advanced Encryption Standard

② DES → Data Encryption Standard

③ IDEA → International Data Encryption Algorithm

④ Blowfish → Drop-in Replacement for DES or IDEA.

⑤ Triple DES

⑥ RC4 (Rivest Cipher 4)

⑦ RC5 (Rivest Cipher 5)

⑧ RC6 (Rivest Cipher 6).

[2] List all Asymmetric Key Algorithm.

→ ① RSA (Rivest Shamir Algorithm)

② Ed25519 Signing

③ DSA (Digital Signature Algorithm)

④ Diffie-Hellman key agreement

⑤ ECC (Elliptic Curve Cryptography)

⑥ El Gamal

⑦ X25519 Key Exchange

⑧ ED448 Signing.

⑨ Key Serialisation.

[3] List the algorithms for message digest.

-> ① MD4 hash

② MD5 hash

③ RIPEMD160 hash

④ SHA-1 hash

⑤ SHA-256 hash

⑥ SHA-512 hash

⑦ NTLM (NT LAN manager)

⑧ LANMAN (LAN Manager

• Network Security •

Page No. 1
Date

• Assignment - 2 •

Q. Describe briefly.

(a) PII (Personally identifiable information)

=> Personally identifiable information defines "Personally" identifiable as information like name, social security number and biometric records, which can be used to distinguish or trace an individual's identity.

(b) U.S. Privacy act of 1974.

=> The privacy act of 1974, Public law 93-579 was created in response to concerns about how the creation and use of computerised databases might impact individuals privacy rights. It safeguards privacy through creating four Procedural and Substantive rights in personal data.

(c) FOIA

=> Freedom of Information Act (FOIA) has established a gateway for americans to access a wealth of data and knowledge maintained by government agencies about virtually anything. And through public-facing websites, citizens are generating FOIA requests at a dizzying pace well more than 55000 annually for the 15 U.S. departments and agencies that receive 90% of all such requests.

(d) FERPA

=> FERPA guarantees students across to their academic records while prohibiting unauthorised access by others. Computer systems maintaining student informa-

tion applicable to FERPA must have computer security controls in place to protect the confidentiality and integrity of this information.

(e) CFAA

-> The computer fraud and abuse Act (CFAA) - title 18 U.S.C. statute 1030 is a law designed to address legal and illegal access to federal and financial I.T. systems. It was intended to reduce the cracking of computer systems and to address federal computer-related offenses.

(f) COPPA

-> The Children's Online Privacy Protection Act of 1998 is a US federal law, located at 15 USC; the act, effective, April 21, 2000, applies to the online collection of personal information by persons or entities under US jurisdiction about children under 13 years of age including children outside of U.S., if the company is U.S. based.

(g) VPPA

-> The video privacy protection act of 1988; an Act to amend title 18, U.S. code, to preserve personal privacy with respect to the rental, purchase or delivery of video tapes or similar audio visual materials.

(h) HIPAA

-> The health Insurance portability and Accountability Act (HIPAA) sets the standard for sensitive patients data protection companies that deal with Protected health information (PHI) must have physical

network and Process security measures in place and follow them to ensure HIPAA compliance.

(i) GLBA

→ The Financial Service Modernisation Act, better known as the Gramm-Leach-Bliley Act (GLBA) requires that financial institutions ensures that the security of customer data, protect data against known or anticipated risks and secure data to protect it from unauthorised access.

(j) PCT DSS.

→ The PCT DSS is an information security standard created to enhance cardholder data security for organisations that store and Process credit credit card data. To attain compliance, organizations must pass an assessment that audits all parts of the network that interact with cardholders environment.

(k) FCRA

→ Faith Credit Reporting Act (FCRA) adds provisions designed to improve the accuracy of consumer's credit related records. The Act also requires the provisions of "risk-based-Pricing" notices & credit scores to consumers in connection with denials or less favourable offers of credit.

(l) FACTA

→ Fair and Accurate Credit Transactions Act (FACTA) is an amendment to FCRA that was added primarily to protect consumers from identity theft. The act stipulated requirements for information privacy, accuracy and disposal and limits the way consumer information can be shared.

• Network Security •

Page No. 1
Date

• Assignment - 2 •

Q State the Full Form for

1) RADIUS
→ Remote Authentication Dial-In User service.

2) TACACS
→ Terminal Access controller Access control System

3) L2TP & PPTP.
→ Layer 2 Tunneling Protocol
Point-to-point tunneling protocol

4) PPP
→ Point-to-Point Protocol

5) EAP
→ Extensible Authentication Protocol

6) CHAP
→ Challenge-Handshake Authentication Protocol

7) NTLM
→ NT (new technology) LAN managers.

8) PAP
→ ~~Putmonary~~ A Password Authentication Protocol

9) SSH
→ Secure Shell

10) LDAP → Lightweight Directory Access Protocol

• Network Security •

Page No. 1
Date

• Assignment - 4 •

[1] List the name & softwares for

(1) Firewall
- Fortigate
- Checkpoint Next Generation Firewalls (NGFWs)
- Sophos XG Firewall
- Watch Guard Network Security
- Sonicwall
- Cisco
- ZoneAlarm
- Glasswire
- Tinywall

(2) Intrusion Detection & Prevention.
- Solar Winds Security Event Manager
- Crowdstrike Falcon
- Manage Engine Eventlog Analyser
- Snort
- OSSEC
- Suricata
- Zeek
- Sagan
- Mcafee Network Security Platform
- Fireeye Network security and forensics.

(3) Anti-virus.
- Bitdefender Antivirus
- Norton Antivirus
- Kaspersky Antivirus
- Avira Antivirus
- Webroot SecureAnyWhere Antivirus

- Avast Antivirus
- Sophos home
- ESET Antivirus.

(4) Packet Sniffing.
- Paessler PRTG network monitor
- Manage Engine NetFlow analyser
- Savious omnipeek
- Wireshark
- Telerik Fiddler
- NETRESEC Network Miner.
- Colasoft Capsa

[2] State any 10 security softwares used by ethical hackers
⇒ (1) Wireshark
(2) Net Stumbler
(3) Burp suite
(4) Aircrack.
(5) Savvius
(6) Web Inspect
(7) Hashcat
(8) Rainbow crack
(9) SQLMap.
(10) Net stumbler
(11) Cain & Abel
etc.

[3] What is the role of CERT?
⇒ CERT - in's primary role is to raise security awareness among indian cyber community and to provide technical assistance and advise them to help them recover from computer

security incidents. CERT-In Provides technical advice to System administrators and users to respond to computers security incidents.

[4] List Top 10 OWASP.

-> - injection
- Broken Authentication
- Sensitive Data Exposure
- XML External Entities (XXE)
- Broken Access control
- Security Misconfigurations
- Cross Site scripting (XSS)
- Insecure Deserialisation
- Using components with known vulnerabilities
- Insufficient logging & monitoring

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y
M.C.A. – 5

R O L L NO : 10
N A M E : Gaurang Sanjaybhai Jani
S U B J E C T : Network Security

R O L L NO : 10
N A M E : Gaurang Sanjaybhai Jani
S U B J E C T : Network Security

	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			

Name – Gaurang Sanjaybhai Jani

Class – MCA-5

Roll no. – 10

Subject – Network Security

Assignment – 1

1.Caesar Cipher

```
import java.util.*;

class CeaserCipher
{

	public static void main (String args[]) throws Exception
	{
		Scanner sc=new Scanner(System.in);
		System.out.print("Enter string to be encrypted : ");
		String ipstr=sc.nextLine();
		int key=6;
		conversion c=new conversion();
		String encrypt=c.convert((key*-1),ipstr);
		System.out.println("Encrypted string : "+encrypt);
		String decrypt=c.convert(key,encrypt);
		System.out.println("Decrypted string : "+decrypt);
	}
}

class conversion
{
```

```
char temp,t;
int ascii=0;
public String convert(int key,String ipstrmsg)
{
	String op="";
	int min=0,max=0,flag=0;
	for(int i=0;i<ipstrmsg.length();i++)
	{
		temp=ipstrmsg.charAt(i);
		ascii=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(ascii>max)
                {
                    int rem=ascii-max;
                    t=(char)((min-1)+rem);
                }
                else if(ascii<min)
                {
                    int rem=min-ascii;
                    t=(char)((max+1)-rem);
                }
                else
                {
                    t=(char)(ascii);
                }

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

Output

```
E:\Gaurang\Study\MCA Sem 5\NS>javac CeaserCipher.java

E:\Gaurang\Study\MCA Sem 5\NS>java CeaserCipher
Enter string to be encrypted : Gaurang
Encrypted string : Auoluha
Decrypted string : Gaurang

E:\Gaurang\Study\MCA Sem 5\NS>
```

2.Substitution Cipher

```
import java.util.*;

class SubstitutionCipher
{
        public static void main (String args[]) throws Exception
        {
            Scanner sc=new Scanner(System.in);
      System.out.print("Enter string to be encrypted : ");
      String ipstr=sc.nextLine();
      System.out.print("Enter value of key : ");
      int key=sc.nextInt();
      conversion c=new conversion();
      String encrypt=c.convert((key*-1),ipstr);
      System.out.println("Encrypted message : "+encrypt);
      String decrypt=c.convert(key,encrypt);
      System.out.println("Decrypted message : "+decrypt);
   }
```

```
}

class conversion
{
        char temp,t;
   int ascii=0;
   public String convert(int key,String ipstrmsg)
   {
            String op="";
      int min=0,max=0,flag=0;
      for(int i=0;i<ipstrmsg.length();i++)
      {
            temp=ipstrmsg.charAt(i);
         ascii=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(ascii>max)
                {
                        int rem=ascii-max;
                        t=(char)((min-1)+rem);
                }
                else if(ascii<min)
                {
                        int rem=min-ascii;
                        t=(char)((max+1)-rem);
                }
                else
                {
                        t=(char)(ascii);
                }

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

```
        }
}
```

Output

```
E:\Gaurang\Study\MCA Sem 5\NS>javac SubstitutionCipher.java

E:\Gaurang\Study\MCA Sem 5\NS>java SubstitutionCipher
Enter string to be encrypted : Gaurang
Enter value of key : 9
Encrypted message : Xrlirex
Decrypted message : Gaurang

E:\Gaurang\Study\MCA Sem 5\NS>
```

3.Transposition Cipher

```
import java.util.*;

class TranspositionCipher

{
        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.println("\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:\Gaurang\Study\MCA Sem 5\NS>javac TranspositionCipher.java

```
E:\Gaurang\Study\MCA Sem 5\NS>java TranspositionCipher
enter key of unique alphabets :
qwertyuiop

enter string :
Gaurang
Cipher text :
u!!!Gragan
Decipher text :
Gaurang
```

4.One Time Pad

```
import java.util.*;

class OneTimePad
{
  public static void main(String args[])throws Exception
  {
    Scanner sc=new Scanner(System.in);
    System.out.println("enter string : ");
    String str1=sc.nextLine();
    char[] pad=new char[str1.length()];
    Random rand = new Random();
    for(int i=0;i<str1.length();i++)
      {
        int ascii=rand.nextInt(123);
        if(ascii<=127)
          {
            pad[i]=(char)ascii;
```

```
        }
       else
        i--;
     }
   System.out.println(pad);
   char[] msg=str1.toCharArray();
   char[] str2=new char[msg.length];
   for(int i=0;i<msg.length;i++)
    {
     str2[i]=(char)(msg[i]^pad[i]);
    }
   System.out.print("cipher text:");
   System.out.print(str2);
   char[] str3=new char[msg.length];
   for(int i=0;i<msg.length;i++)
    {
     str3[i]=(char)(pad[i]^str2[i]);
    }
   System.out.print("\noriginal text:");
   System.out.print(str3);
  }
}
```

Output

```
E:\Gaurang\Study\MCA Sem 5\NS>javac OneTimePad.java

E:\Gaurang\Study\MCA Sem 5\NS>java OneTimePad
enter string :
hello world!
wO→FGb.o.→w%
```

cipher text:▼*v*(BY \v!!♦

original text:hello world!

5.Port Scanner

```
import java.net.*;

public class PortScanner
{
   public static void main(String args[])
   {
      for(int i=1; i<=65535; i++)
      {
         try
         {
            Socket s=new Socket();
            s.connect(new InetSocketAddress("localhost",i),1000);
            s.close();
            System.out.println("Port "+i+" is open!");
         }
         catch(Exception e)
         { }
      }
   }
}
```

Output

```
E:\Gaurang\Study\MCA Sem 5\NS>javac PortScanner.java

E:\Gaurang\Study\MCA Sem 5\NS>java PortScanner
Port 25 is open!
```

Port 80 is open!

Port 110 is open!

Port 119 is open!

Port 135 is open!

Port 143 is open!

6.P-Box

```
import java.util.Scanner;

class PBox{
  public String doEncryption(String s)
  {
    byte p[]=new byte[8];
    byte pTemp[]=new byte[8];
    pTemp=s.getBytes();
    p[0]=pTemp[1];
    p[1]=pTemp[7];
    p[2]=pTemp[0];
    p[3]=pTemp[5];
    p[4]=pTemp[6];
    p[5]=pTemp[2];
    p[6]=pTemp[3];
    p[7]=pTemp[4];
    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[2];
    p[1]=pTemp[0];
    p[2]=pTemp[5];
    p[3]=pTemp[6];
    p[4]=pTemp[7];
    p[5]=pTemp[3];
    p[6]=pTemp[4];
    p[7]=pTemp[1];
    return(new String(p));
  }
  public static void main(String args[]){
    Scanner sc=new Scanner(System.in);
    System.out.println("Enter string of 8 characters : ");
    String plaintext=sc.nextLine();
    PBox pbox=new PBox();
    System.out.println("Encrypted Text : " + pbox.doEncryption(plaintext));
    System.out.println("Decrypted Text : " +
pbox.doDecryption(pbox.doEncryption(plaintext)));
  }
}
```

Output

```
E:\Gaurang\Study\MCA Sem 5\NS>javac PBox.java

E:\Gaurang\Study\MCA Sem 5\NS>java PBox
Enter string of 8 characters :
ABCDEFGH
Encrypted Text : BHAFGCDE
Decrypted Text : ABCDEFGH

E:\Gaurang\Study\MCA Sem 5\NS>
```

7.S-Box

```
import java.util.*;


class SBox{
  char key[][];
  Random rand;
  SBox()
  {
    rand=new Random();
    int add=rand.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

```
E:\Gaurang\Study\MCA Sem 5\NS>javac SBox.java

E:\Gaurang\Study\MCA Sem 5\NS>java SBox
Enter Message :
Hello World!
Encrypted Text : LippsAsvph
Decrypted Text : HelloWorld
```

E:\Gaurang\Study\MCA Sem 5\NS>

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

```
E:\Gaurang\Study\MCA Sem 5\NS>javac DES.java

E:\Gaurang\Study\MCA Sem 5\NS>java DES
Enter Message :
Hello World!
Encrypted Text : 8◄?♠_?mí1Xr£ç?X:
Encrypted Text : Hello World!

E:\Gaurang\Study\MCA Sem 5\NS>
```

9.AES

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

class AES
{
    private byte[] key;
    AES()
    {
        key="qwertyuiop123456".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

```
E:\Gaurang\Study\MCA Sem 5\NS>javac AES.java

E:\Gaurang\Study\MCA Sem 5\NS>java AES
Enter Message :
This is AES example
```

Encrypted Text : i↕¡Q?É☻?♀=???k6ë?Ä+¥F♥?_???8

Encrypted Text : This is AES example

E:\Gaurang\Study\MCA Sem 5\NS>

10.RSA

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;

class RSA
{
   public KeyPairGenerator keygen;
   public KeyPair myKey;
   Cipher ciph;
   public RSA() throws Exception
   {
      keygen = KeyPairGenerator.getInstance("RSA");
      keygen.initialize(512) ;
      myKey = keygen.generateKeyPair();
      ciph = Cipher.getInstance("RSA");
   }
   public byte[] doEncryption(String s) throws Exception
   {
      ciph.init(Cipher.ENCRYPT_MODE,myKey.getPublic());
      byte[] text = s.getBytes();
      byte[] textEncrypted = ciph.doFinal(text);
      return(textEncrypted);
   }
```

```
    public String doDecryption(byte[] s)throws Exception
    {
        ciph.init(Cipher.DECRYPT_MODE,myKey.getPrivate());
        byte[] textDecrypted = ciph.doFinal(s);
        return(new String(textDecrypted));
    }
    public static void main(String[] argv) throws Exception
    {
        RSA d=new RSA();
        byte[] str=d.doEncryption("GaurangJani");
        System.out.println("Encrypted String : "+str);
        System.out.println("Decrypted String : "+d.doDecryption(str));
    }
}
```

Output

```
E:\Gaurang\Study\MCA Sem 5\NS>javac RSA.java

E:\Gaurang\Study\MCA Sem 5\NS>java RSA
Encrypted String : [B@22a71081
Decrypted String : GaurangJani

E:\Gaurang\Study\MCA Sem 5\NS>
```

11.SHA

```
import java.util.Scanner;
import java.math.*;
import java.security.*;

class SHA
```

```
{
   private String doEncryption(String str) throws Exception
   {
      MessageDigest md=MessageDigest.getInstance("SHA-1");
      byte[] msg=md.digest(str.getBytes());
      BigInteger bigInt=new BigInteger(1,msg);
      String hashValue=bigInt.toString(32);
      while(hashValue.length()<64)
         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 str=sc.nextLine();
      System.out.println("Hash Text : " + sha.doEncryption(str));
   }
}
```

Output

```
E:\Gaurang\Study\MCA Sem 5\NS>javac SHA.java

E:\Gaurang\Study\MCA Sem 5\NS>java SHA
Enter Message :
Alpha Bravo Charlie Delta Echo Foxtrot Golf Hotel
Hash Text : fgdqn2emju5opt4f342okm34ft49159u0fgdqn2emju5opt4f342okm34ft49159u
```

E:\Gaurang\Study\MCA Sem 5\NS>

12.MD5

```
import java.util.Scanner;
import java.math.*;
import java.security.*;

class MD5
{
   private String doEncryption(String text) throws Exception
   {
      MessageDigest msgdgst=MessageDigest.getInstance("MD5");
      byte[] msg=msgdgst.digest(text.getBytes());
      BigInteger bigInt=new BigInteger(1,msg);
      String hashValue=bigInt.toString(32);
      while(hashValue.length()<64)
         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

E:\Gaurang\Study\MCA Sem 5\NS>javac MD5.java

E:\Gaurang\Study\MCA Sem 5\NS>java MD5

Enter Message :

Alpha Bravo Charlie Delta Echo Foxtrot Golf

Hash Text : vrgosv7pumkbkcd4roelhvgre0vrgosv7pumkbkcd4roelhvgre0vrgosv7pumkbkcd4

roelhvgre0vrgosv7pumkbkcd4roelhvgre

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)

Prog13_Client.java

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

class Prog13_Client
{
   public static void main(String args[]) throws IOException
   {
      DatagramSocket client = new DatagramSocket();
      InetAddress ip = InetAddress.getLocalHost();
      byte[] send = new byte[65536];
      byte[] recieve = new byte[65536];
      Scanner in = new Scanner(System.in);
      System.out.print("\nEnter username:");
```

```
    String username=in.nextLine();
    System.out.print("\nEnter password:");
    String password = in.nextLine();
    password = encryptPassword(password);
    String data=username+","+password;
    send=data.getBytes();
    DatagramPacket packet = new DatagramPacket(send,send.length,ip,6969);
    client.send(packet);
    packet = new DatagramPacket(recieve,recieve.length);
    client.receive(packet);
    System.out.print("Server:"+convertToString(recieve));
}

public static String convertToString(byte[] a)
{
    if (a == null)
        return null;
    String s = "";
    int i = 0;
    while (a[i] != 0)
    {
        s=s+(char)a[i];
        i++;
    }
    return s;
}
public static String encryptPassword(String password)
{
    char[] tokens = password.toCharArray();
    String encrypted_text="";
```

```
        for(char c:tokens)
        {
            encrypted_text=encrypted_text+((char)((int)c+3))+"";
        }
        return encrypted_text;
    }
}
```

Prog13_Server.java

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

class Prog13_Server
{
    static HashMap<String,String> user_data = new HashMap<String,String>();
    public static void main(String args[]) throws IOException
    {
        user_data.put("Gaurang","Gaurang");
        user_data.put("MCA510","MCA510");
        DatagramSocket server = new DatagramSocket(6969);
        byte[] send = new byte[65536];
        byte[] recieve = new byte[65536];
        DatagramPacket packet = new DatagramPacket(recieve,recieve.length);
        server.receive(packet);
        String[] message=(convertToString(recieve)).split(",");
        String username=message[0];
        String password=decryptPassword(message[1]);
        InetAddress ip = packet.getAddress();
        int port=packet.getPort();
```

```
    System.out.print("\nUsername : "+username+"\nPassword : "+password);
    String status=check(password,username);
    packet=new DatagramPacket(status.getBytes(),status.getBytes().length,ip,port);
    server.send(packet);
}
public static String check(String password,String username)
{
    if(password.equals((String)user_data.get(username)))
    {
        return "success";
    }
    return "failure";
}
public static String convertToString(byte[] a)
{
    if (a == null)
        return null;
    String s = "";
    int i = 0;
    while (a[i] != 0)
    {
        s=s+(char)a[i];
        i++;
    }
    return s;
}
public static String decryptPassword(String password)
{
    char[] tokens = password.toCharArray();
    String decrypted_text="";
```

```
        for(char c:tokens)
        {
            decrypted_text=decrypted_text+((char)((int)c-3))+"";
        }
        return decrypted_text;
    }
}
```

Output

Server Side

```
E:\Gaurang\Study\MCA Sem 5\NS>javac Prog13_Client.java

E:\Gaurang\Study\MCA Sem 5\NS>javac Prog13_Server.java

E:\Gaurang\Study\MCA Sem 5\NS>java Prog13_Server

Username : Gaurang
Password : Gaurang
E:\Gaurang\Study\MCA Sem 5\NS>java Prog13_Server

Username : ABC
Password : DEF
E:\Gaurang\Study\MCA Sem 5\NS>
```

Client Side

```
E:\Gaurang\Study\MCA Sem 5\NS>javac Prog13_Client.java
```

E:\Gaurang\Study\MCA Sem 5\NS>javac Prog13_Server.java

E:\Gaurang\Study\MCA Sem 5\NS>java Prog13_Server

Username : Gaurang

Password : Gaurang

E:\Gaurang\Study\MCA Sem 5\NS>java Prog13_Server

Username : ABC

Password : DEF

E:\Gaurang\Study\MCA Sem 5\NS>

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).

Prog14_Server.java

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

class Prog14_Server
{
   public static void main(String args[])throws Exception
   {
      ServerSocket ss=new ServerSocket(6969);
      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/mydb?useUnicode=true&useJD
BCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC",
"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("Login Successful");
        }
        else
        {
           dos.writeUTF("Incorrect Password");
        }
    }
}



class Conversion_Server
```

```
{
    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;
    }
}
```

Prog14_Client.java

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

class Prog14_Client
{
    public static void main(String args[])throws Exception
    {
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter username : ");
        String uname=sc.nextLine();
        System.out.print("Enter password : ");
        String pwd=sc.nextLine();
        Socket s=new Socket("127.0.0.1",6969);
        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_Client
{
    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

```
E:\Gaurang\Study\MCA Sem 5\NS>javac Prog14_Client.java

E:\Gaurang\Study\MCA Sem 5\NS>java Prog14_Client
Enter username : Gaurang
Enter password : Gaurang
Login Successful
```

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

Prog15_Client.java

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

class Prog15_Client
{
   public static void main(String args[])throws Exception
   {
      Scanner sc=new Scanner(System.in);
      System.out.print("Enter packet to be sent to server : ");
```

```
        String packet=sc.nextLine();
        Socket s=new Socket("127.0.0.1",6969);
        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();
    }
}
```

Prog15_Firewall.java

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

class Prog15_Firewall
{
    public static void main(String args[])throws Exception
    {
        ServerSocket ss=new ServerSocket(6969);
        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="Package contains threat, cannot be delivered";
      dos.writeUTF(err);
      s1.close();
      flag=1;
      break;
   }
}
if(flag==0)
{
   Socket s2=new Socket("127.0.0.1",9696);
   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 has been recieved by the server");
   }
   else
   {
      dos.writeUTF("Server unavailable");
   }
   s1.close();
   s2.close();
}
```

```
        ss.close();
    }
}
```

Prog15_Server.java

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

class Prog15_Server
{
    public static void main(String args[])throws Exception
    {
        ServerSocket ss=new ServerSocket(9696);
        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

Successful

Client

```
E:\Gaurang\Study\MCA Sem 5\NS>java Prog15_Client
```

Enter packet to be sent to server : Gaurang

packet has been recieved by the server

E:\Gaurang\Study\MCA Sem 5\NS>

Server

E:\Gaurang\Study\MCA Sem 5\NS>java Prog15_Server

client packet:Gaurang

Failed

Client

E:\Gaurang\Study\MCA Sem 5\NS>Java Prog15_Client

Enter packet to be sent to server : terrorist

Package contains threat, cannot be delivered

E:\Gaurang\Study\MCA Sem 5\NS>

Server

E:\Gaurang\Study\MCA Sem 5\NS>java Prog15_Server

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

Prog16_Client.java

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

```
class Prog16_Client
{
   public static void main(String args[])throws Exception
   {
      Scanner sc=new Scanner(System.in);
      String numbers="";
      System.out.print("Enter set of numbers to be sent :");
      String num=sc.nextLine();
      Socket s=new Socket("127.0.0.1",6969);
      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();
   }
}
```

Prog16_Forwarder.java

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

class Prog16_Forwarder
{
   public static void main(String args[])throws Exception
   {
      ServerSocket ss=new ServerSocket(6969);
      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",9696);
Socket s2=new Socket("127.0.0.1",9393);
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();
    }
}
```

Prog16_ServerOdd.java

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

class Prog16_ServerOdd
{
 public static void main(String args[])throws Exception
 {
  ServerSocket ss=new ServerSocket(9696);
  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();
 }
}
```

Prog16_ServerEven.java

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

class Prog16_ServerEven
{
    public static void main(String args[])throws Exception
    {
        ServerSocket ss=new ServerSocket(9393);
        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

Client

```
E:\Gaurang\Study\MCA Sem 5\NS>java Prog16_Client
Enter set of numbers to be sent :1 2 3 4 5 6 7 8
packets delivered to servers

E:\Gaurang\Study\MCA Sem 5\NS>
```

ServerOdd

```
E:\Gaurang\Study\MCA Sem 5\NS>java Prog16_ServerOdd
client packet: 1 3 5 7

E:\Gaurang\Study\MCA Sem 5\NS>
```

ServerEven

E:\Gaurang\Study\MCA Sem 5\NS>java Prog16_ServerEven

client packet: 2 4 6 8

E:\Gaurang\Study\MCA Sem 5\NS>

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.

Prog17_Client.java

```
import java.io.DataInputStream;
import java.net.Socket;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;


class Prog17_Client
{
   static String receiverid;
   static SecretKeySpec receiverkey;
   public static void main(String args[]) throws Exception
   {
      System.out.println("client");
      receiverid="receiver1";
      receiverkey=new SecretKeySpec("36369696".getBytes(),"DES");
      Socket s=new Socket("localhost",6969);
```

```
        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("receiver id : " +new String(receiverid));

        byte[] sessionkey=cipher.doFinal(encryptedsessionkeyclient);
        System.out.println("session key : " + new String(sessionkey));
    }
}
```

KDC.java

```
import java.io.*;
import java.net.*;
import java.security.*;
import javax.crypto.*;
```

```
class KDC
{
   public static void main(String args[]) throws Exception
   {
      SecretKeySpec senderkey,receiverkey;
      byte [] sessionkey,encryptedsessionkey;
      String senderid,receiverid;
      System.out.println("KDC");
      receiverid="receiver1";
      senderid="sender1";
      receiverkey=new SecretKeySpec("36369696".getBytes(),"DES");
      senderkey=new SecretKeySpec("69696363".getBytes(),"DES");
      ServerSocket ss=new ServerSocket(9696);
      Socket s=ss.accept();
      sessionkey=generateSessionKey();
      System.out.println("session key : " +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;
    }
}
```

Prog17_Server.java

```
import java.io.*;
import java.net.*;
import javax.crypto.*;

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="sender1";
        senderkey=new SecretKeySpec("69696363".getBytes(),"DES");
        getSessionInfoServer();
        ServerSocket ss=new ServerSocket(6969);
```

```
        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(),9696);
        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("server session key : " +new String(sessionkey));
    }
}
```

Output

Server Side

E:\Gaurang\Study\MCA Sem 5\NS>javac Prog17_Server.java KDC.java Prog17_Client.java

E:\Gaurang\Study\MCA Sem 5\NS>java Prog17_Server

Server

server session key : A��|�^

Client Side

---

sender id : sender1

receiver id : receiver1

session key : A��|�^

KDC

---

session key : A��|�^