CYBER
&
INFORMATION SECURITY

INDEX

Practical 1

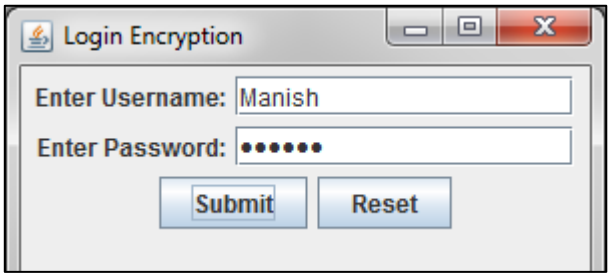
Write a program to store username and password in an encrypted form in a database to implement integrity lock.

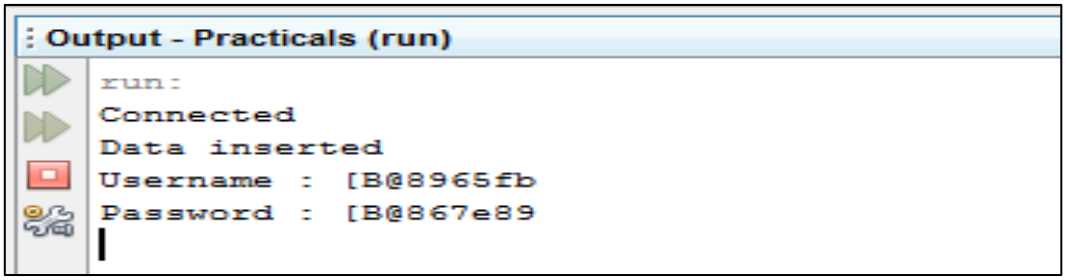
Code:

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.sql.*;
import javax.crypto.*;
public class LoginFinal extends JFrame implements ActionListener{
   JLabel lblUser, lblPass;
   JTextField txtUser;
   JPasswordField txtPass;
   JButton btnSub, btnRes;
   Connection con;
   PreparedStatement pst;
   Container c;
   SecretKey key;
   Cipher cipher;
   public LoginFinal()    {
      c = getContentPane();
      lblUser = new JLabel("Enter Username:");
      txtUser = new JTextField(15);
      lblPass = new JLabel("Enter Password:");
      txtPass = new JPasswordField(15);
      btnSub = new JButton("Submit");
      btnRes = new JButton("Reset");
      c.add(lblUser);
      c.add(txtUser);
      c.add(lblPass);
      c.add(txtPass);
      c.add(btnSub);
      c.add(btnRes);
      c.setLayout(new FlowLayout());
      setVisible(true);
      setSize(300,200);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      setTitle("Login Encryption");
      btnSub.addActionListener(this);
      btnRes.addActionListener(this);
      try       {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
         con = DriverManager.getConnection("jdbc:odbc:LoginDSN");
         System.out.println("Connected");
         cipher = Cipher.getInstance("DESede");
         key = KeyGenerator.getInstance("DESede").generateKey();
      }
      catch(Exception e)
```

```
   {e.printStackTrace(); }}
  public void actionPerformed(ActionEvent ae) {
    if(ae.getSource()==btnSub)  {
      try   {
        String username = txtUser.getText();
        String password = txtPass.getText();
        byte[] uname = encrypt(username);
        byte[] pass = encrypt(password);
        String query = "insert into Login values(?,?)";
        pst=con.prepareStatement(query);
        pst.setBytes(1, uname);
        pst.setBytes(2, pass);
        pst.executeUpdate();
        System.out.println("Data inserted");
        System.out.println("Username : " +uname);
        System.out.println("Password : " +pass);
      }
      catch(SQLException e)  {
        e.printStackTrace();
      } }
    if(ae.getSource()==btnRes)      {
      txtUser.setText("");
      txtPass.setText("");
    }}
  public byte[] encrypt(String data)   {
    byte[] encryptedString = null;
    try      {
      cipher.init(Cipher.ENCRYPT_MODE,key);
      encryptedString = cipher.doFinal(data.getBytes());
    }
    catch(Exception e) {
      e.printStackTrace();
    }
    return encryptedString;
  }
  public static void main(String[] arg)   {
    new LoginFinal();   }}
```

Output:


Login Encryption
Enter Username:
Manish
Enter Password:
Submit
Reset


Output - Practicals (run)
run:
Connected
Data inserted
Username : [B@8965fb
Password : [B@867e89

Practical 2

Write SQL query to retrieve sensitive information from less sensitive queries.

Code:

```
SQL> create table mytable
  2    (
  3    name varchar2(20),
  4    sex varchar2(10),
  5    race varchar2(10),
  6    aid number(10),
  7    fines varchar2(10),
  8    drugs number(5),
  9    dorm varchar2(10)
 10   );

Table created.
SQL> insert into mytable values('Adams','M','C',5000,'45',1,'Holmes');

1 row created.

SQL> insert into mytable values('Fein','F','C',1000,'15',0,'West');

1 row created.

SQL> insert into mytable values('Groff','M','C',4000,'0',3,'West');

1 row created.

SQL>  insert into mytable values('Hill','F','V',5000,'10',2,'Holmes');

1 row created.

SQL> insert into mytable values('Koch','F','C',0,'0.',1,'West');

1 row created.

SQL>  insert into mytable values('Liu','F','A',0,'10',2,'Grey');

1 row created.

SQL>  insert into mytable values('Majors','M','C',2000,'0',2,'Grey');

1 row created.

SQL>  insert into mytable values('Hill','F','V',5000,'10',2,'Holmes');

1 row created.

SQL> insert into mytable values('Koch','F','C',0,'0.',1,'West');
```

1 row created.

SQL> insert into mytable values('Liu','F','A',0,'10',2,'Grey');

1 row created.

SQL> insert into mytable values('Majors','M','C',2000,'0',2,'Grey');

1 row created.

Select * from mytable;

NAME	SEX	RACE	AID	FINES	DRUGS	DORM
Adams	M	C	5000	45	1	Holmes
Bailey	M	B	0	0	0	Grey
Chin	F	A	3000	20	0	West
Dewitt	M	B	1000	35	3	Grey
Earhart	F	C	2000	95	1	Holmes
Fein	F	C	1000	15	0	West
Groff	M	C	4000	0	3	West
Hill	F	V	5000	10	2	Holmes
Koch	F	C	0	0	1	West
Liu	F	A	0	10	2	Grey
Majors	M	C	2000	0	2	Grey

1] Direct attack
In the given table, financial aid and drug are sensitive columns..

1)Display the name of the male with drug count 1.

SQL> select name from mytable where sex='M' and drugs=1;

NAME

Adams

Display the name where drug count is 1 or dorm is West

```
SQL> select name from mytable where (sex='M' and drugs=1) or (sex='M' and sex='F')     or (dorm='west');


NAME
--------------------
Adams
```

2] Indirect attack

Sum attack

```
SQL>  create table female_aid as (select sum(aid) as f_aid, dorm from mytable where sex='F'      group by dorm);

Table created.

SQL> create table male_aid as (select sum(aid) as m_aid, dorm from mytable where
 sex='M' group by dorm);

Table created.

SQL>  select m.dorm, m.m_aid, f.f_aid, m.m_aid+f.f_aid as total from female_aid
f, male_aid m where m.dorm(+)=f.dorm;



DORM      M_AID     F_AID     TOTAL
----------    ----------    ----------   ----------
Holmes       5000      10000     15000
Grey         4000        0         4000
West          4000      1000       5000
```

Alternate way:

```
SQL>  select f.dorm,f.female,m.male,m.male+f.female as Total from  (select sum(a
id) As Female,dorm from mytable  where sex='F' group  by dorm) f,(select sum(aid
) As Male,dorm from mytable where  sex='M' group by dorm) m where m.dorm=f.dorm;

DORM        FEMALE    MALE     TOTAL
----------       ----------      ----------    ----------
Holmes        10000         5000        15000
Grey            0             4000        4000
West           1000          4000        5000
```

Count attack

```
SQL>  create table female as (select count(*) as female, dorm from mytable where
 sex='F' group by dorm);

Table created.

SQL> create table male as (select count(*) as male, dorm from mytable where sex=
'M' group by dorm);

Table created.

SQL> select f.dorm, m.male, f.female,  m.male + f.female as total from male m, f
emale f where m.dorm(+)=f.dorm;

DORM      MALE    FEMALE     TOTAL
----------  ---------- ----------  ----------
Holmes        1        2              3
Grey          2        2              4
West          1        3              4
```

Alternative way:

```
SQL> select f.dorm,f.female,m.male,m.male+f.female as Total from  (select count(
*) As Female,dorm from mytable  where sex='F' group  by dorm) f,(select count(*)
 As Male,dorm from mytable where  sex='M' group by dorm) m where m.dorm=f.dorm;

DORM        FEMALE     MALE     TOTAL
----------          ----------       ---------- ----------
Holmes          2                1       3
Grey            2                2       4
West            3                1       4
```

Median attack

a) Conduct a median attack to get the aid of males.

```
SQL> select PERCENTILE_CONT(0.5) within group
          2             (order by aid) as perc_disc from mytable where sex='M';

 PERC_DISC
----------
3000
```

b) Conduct a median attack to get the aid with drug count 2.

```
    SQL> select PERCENTILE_CONT(0.5) within group
2     (order by aid) as perc_disc from mytable where drugs=2;
PERC_DISC
----------
2000
```

Practical 3

Write SQL query to create a view implement concept of views and commutative filter in distributed databases.

Code:

VERTICAL FRAGMENTATION

System :

```
create table emp_glo
   (
   eno number(5),
   ename varchar2(40),
   address varchar2(50),
   email varchar2(40),
   sal number(7,2)
   );
```

Table created.

Node1:(usr111)

```
create table emp1
  2  (
  3  eno number(5),
  4  ename varchar2(40),
  5  address varchar2(50)
  6  );
```

Table created.

Node2(usr222):

```
create table emp2
  2  (
  3  eno number(5),
  4  email varchar2(40),
  5  sal number(7,2)
  6  );
```

System:

```
create database link link1 connect to usr111 identified by "usr111" using 'orcl';
create database link link2 connect to usr222 identified by "usr222" using 'orcl';
```

Trigger:

```
SQL> create or replace trigger trigEmp_glo
  2     after insert on emp_glo
  3     for each row
  4     begin
  5     insert into emp1@link1 values(:new.eno, :new.ename, :new.address);
  6     insert into emp2@link2 values(:new.eno, :new.email, :new.sal);
  7     end;
  8  /
```

Trigger created.

```
SQL> insert into emp_glo values(1,'Dev','Byculla','d@gmail.com',50000);

1 row created.

SQL> insert into emp_glo values(2,'Rina','Dombivli','r@gmail.com',60000);

1 row created.

SQL>  insert into emp_glo values(3,'Abhay','Kalyan','a@gmail.com',40000);

1 row created.

SQL>  insert into emp_glo values(4,'Deepali','Dombivli','dpl@gmail.com',30000);

1 row created.

SQL>  insert into emp_glo values(5,'Kalpana','Thane','k@gmail.com',30000);

1 row created.

select * from emp_glo;
    ENO         ENAME          ADDRESS         EMAIL              SAL
---------- ------------------------------------------------------    -------------  ----------
      1             Dev               Byculla         d@gmail.com     50000
      2             Rina                Dombivli      r@gmail.com      60000
      3             Abhay               Kalyan        a@gmail.com     40000
      4             Deepali    Dombivli      dpl@gmail.com  30000
      5             Kalpana             Thane         k@gmail.com     30000

Select * from emp1@link1;

ENO     ENAME          ADDRESS
---------- ------------------------------------------------------------------------------------------
      1   Dev                       Byculla
      2   Rina                      Dombivli
      3   Abhay           Kalyan
      4   Deepali                   Dombivli
      5   Kalpana                   Thane
SQL> select * from emp2@link2;

   ENO   EMAIL                              SAL
---------- ----------------------------------------    ----------
      1   d@gmail.com                        50000
      2   r@gmail.com                        60000
      3   a@gmail.com                        40000
      4   dpl@gmail.com                      30000
      5   k@gmail.com                        30000
```

Q. Create view to display name and salary of emp whose salary is greater than avg salary of emp.

```
SQL> create view ViewSal
  2  as
  3   select e1.ename, e2.sal
  4  from emp1@link1 e1, emp2@link2 e2
  5  where e1.eno=e2.eno and e2.sal>(select avg(sal) from emp2@link2);
```

View created.

```
SQL> select * from ViewSal;

ENAME                          SAL
---------------------------------------- ----------
Dev                            50000
Rina                           60000
```

HORIZONTAL FRAGMENTATION

System :

```
SQL>  create table emp_glo2
  2    (
  3     eno number(5),
  4     ename varchar2(40),
  5     address varchar2(50),
  6     email varchar2(40),
  7     sal number(7,2)
  8     )
  9  ;
```

Table created.

Node1(usr111):

```
SQL>  create table emp101
  2     (
  3     eno number(5) primary key,
  4     ename varchar2(40),
  5     address varchar2(50),
  6     email varchar2(40),
  7     sal number(7,2)
  8     );
```

Table created.

Node2(usr222):

```
SQL> create table emp102
  2     (
  3     eno number(5) primary key,
```

```
 4    ename varchar2(40),
 5    address varchar2(50),
 6    email varchar2(40),
 7    sal number(7,2)
 8   );
```

Table created.

System:

Trigger:

```
SQL>  create or replace trigger triginsertemp102
 2    after insert on emp_glo2
 3    for each row
 4    begin
 5    if :new.sal>10000 and :new.sal<=20000
 6    then
 7         insert into emp102@link2 values(:new.eno,:new.ename,:new.address ,:new.email,:new.sal);
 8    else
 9         insert into emp101@link1 values(:new.eno,:new.ename,:new.address,:new.email,:new.sal);
10    end if;
11    end;
12    /
```

Trigger created.

Inserting Records

```
SQL> insert into emp_glo2 values(1,'Dev','Byculla','d@gmail.com',30000);
```

1 row created.

```
SQL>  insert into emp_glo2 values(2,'Rina','Dombivli','r@gmail.com',35000);
```

1 row created.

```
SQL> insert into emp_glo2 values(3,'Abhay','Kalyan','a@gmail.com',20000);
```

1 row created.

```
SQL> insert into emp_glo2 values(4,'Deepali','Dombivli','dpl@yahoo.com',16000);
```

1 row created.

```
SQL>  insert into emp_glo2 values(5,'Kalpana','Thane','k@yahoo.com',15000);
```

1 row created.

Displaying data:

```
SQL> select * from emp_glo2;
```

```
ENO         ENAME         ADDRESS      EMAIL                SAL
---------- ------------------------------------------------------------------------------------------
      1      Devendra    Byculla d@gmail.com          30000
      2      Ritesh              Dombivli       r@gmail.com          35000
      3      Abhay               Kalyan         a@gmail.com          20000
      4      Deepal              Dombivli       dpl@yahoo.com        16000
      5      Kalps               Thane          k@yahoo.com          15000

SQL> select * from emp101@link1;
    ENO     ENAME     ADDRESS        EMAIL          SAL
---------- ---------- ------------------------------------------------------------------------------ ------
      1          Dev     Byculla          d@gmail.com    30000
      2          Rina    Dombivli         r@gmail.com     35000

SQL> select * from emp102@link2;

    ENO         ENAME       ADDRESS          EMAIL                SAL
---------- ---------- ------------------------------------------------------------------------------ ----------
      3          Abhay          Kalyan       a@gmail.com          20000
      4          Deepali        Dombivli     dpl@yahoo.com        16000
      5          Kalpana        Thane         k@yahoo.com         15000
```

Q. Create view to display emails of the employees who don't use Gmail.

```
SQL> create view viewEmail as select email from emp102@link2 where email not like '%gmail.com';

View created.

SQL> select * from viewEmail;

EMAIL
----------------------------------------
dpl@yahoo.com
k@yahoo.com
```

Practical 4

Write a program to implement SSL.

SslSv.java

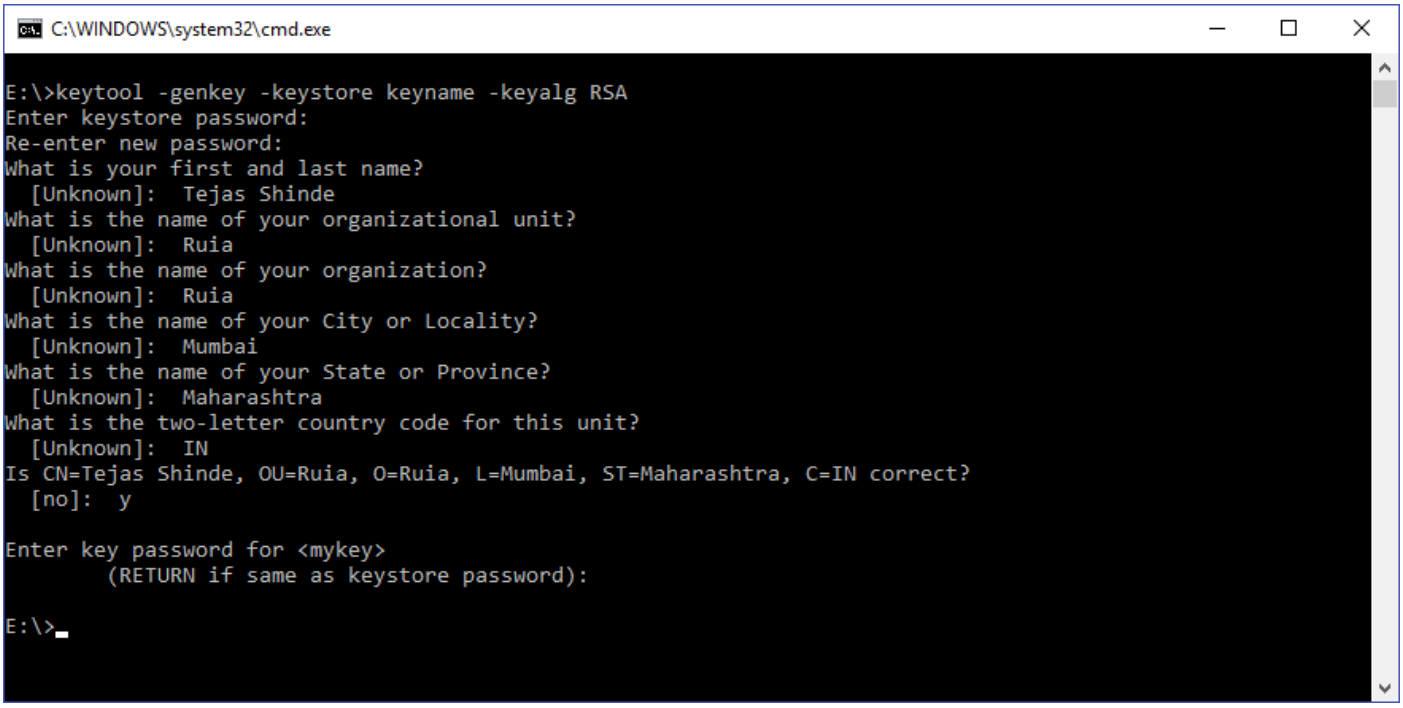
```
import java.io.*;
import javax.net.ssl.*;
public class SslSv {
   public static void main(String[] args) {
      try {
         SSLServerSocketFactory sf = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault();
         SSLServerSocket ss = (SSLServerSocket) sf.createServerSocket(6017);
         SSLSocket socket = (SSLSocket) ss.accept();
String str;
         BufferedReader br = new BufferedReader(new InputStreamReader(socket.getInputStream()));
         while ((str=br.readLine()) != null) {
            System.out.println(str);
            System.out.flush();
         } } catch (Exception e) {
         e.printStackTrace();
      }}}
```

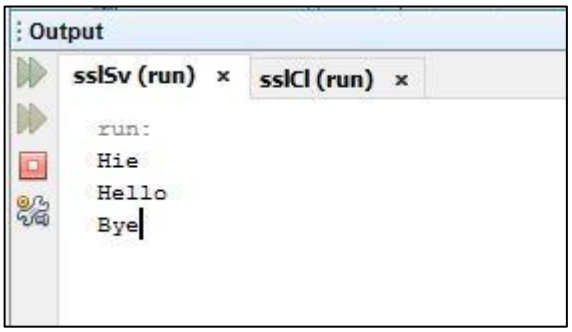
SslCl.java

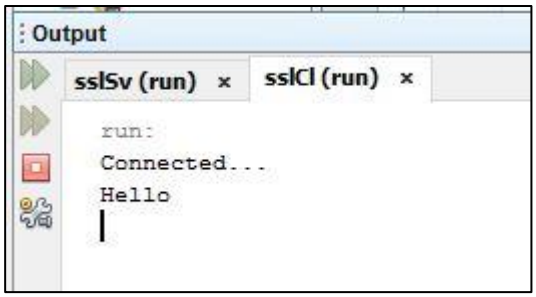
```
import java.io.*;
import javax.net.ssl.*;
public class SslCl {
   public static void main(String[] args) {
      try {
         SSLSocketFactory sf = (SSLSocketFactory) SSLSocketFactory.getDefault();
         SSLSocket sk = (SSLSocket) sf.createSocket("localhost", 6017);
         System.out.println("Connected...");
         BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
         OutputStreamWriter osw = new OutputStreamWriter(sk.getOutputStream());
         BufferedWriter bw = new BufferedWriter(osw);
         String str = null;
         while ((str = br.readLine()) != null) {
            bw.write(str);
            bw.flush();
         }
      } catch (Exception e) {
         e.printStackTrace();
      }}}
```

Output:

Note: Create a ssl certificate using following command.







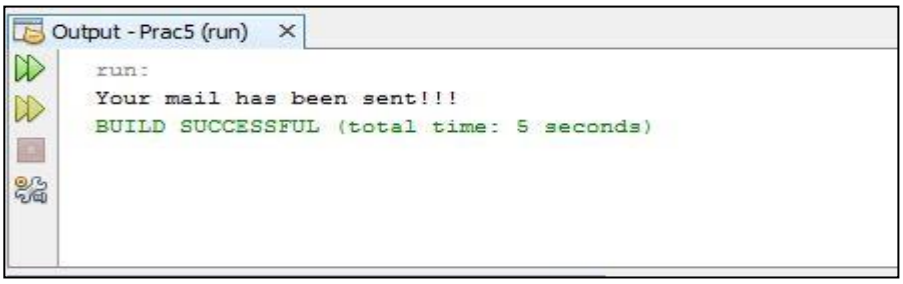
Practical 5

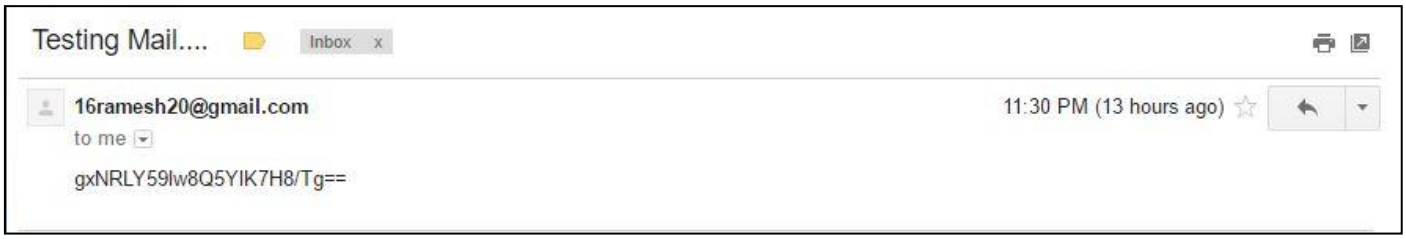
Write a program to send an encrypted email.

Code:

```
import java.util.Properties;
import javax.crypto.*;
import javax.mail.*;
import javax.mail.internet.*;
import sun.misc.BASE64Encoder;
public class Prac5 {
   public static void main(String[] args) {
      Properties props = new Properties();
      props.put("mail.smtp.starttls.enable", "true");
      props.put("mail.smtp.host", "smtp.gmail.com");
      props.put("mail.smtp.ssl.trust", "smtp.gmail.com");
      props.put("mail.smtp.socketFactory.port", "465");
      props.put("mail.smtp.socketFactory.class","javax.net.ssl.SSLSocketFactory");
      props.put("mail.smtp.auth", "true");
      props.put("mail.smtp.port", "465");
      Session session = Session.getDefaultInstance(props,
           new javax.mail.Authenticator() {
        protected PasswordAuthentication getPasswordAuthentication() {
          return new PasswordAuthentication("16ramesh20@gmail.com", "password");
        }}); try {
        String msg = "hello all!!!", cipherText, decryptedText;
        KeyGenerator keyGen = KeyGenerator.getInstance("AES");
        keyGen.init(128);
        SecretKey secretKey = keyGen.generateKey();
        Cipher aesCipher = Cipher.getInstance("AES");
        aesCipher.init(Cipher.ENCRYPT_MODE, secretKey);
        byte[] byteDataToEncrypt = msg.getBytes();
        byte[] byteCipherText = aesCipher.doFinal(byteDataToEncrypt);
        cipherText = new BASE64Encoder().encode(byteCipherText);
        Message message = new MimeMessage(session);
        message.setFrom(new InternetAddress("16ramesh20@gmail.com"));
        message.setRecipients(Message.RecipientType.TO,
             InternetAddress.parse("someone23@gmail.com"));
        message.setSubject("Testing Mail....");
        message.setText(cipherText);
        Transport.send(message);
        System.out.println("Your mail has been sent!!!");
      } catch (Exception e) {
        System.out.println(e);
      }}}
```

Output:


Output - Prac5 (run)
run:
Your mail has been sent!!!
BUILD SUCCESSFUL (total time: 5 seconds)


Testing Mail....
Inbox x
16ramesh20@gmail.com
11:30 PM (13 hours ago)
to me
gxNRLY59lw8Q5YIK7H8/Tg==

Practical 6

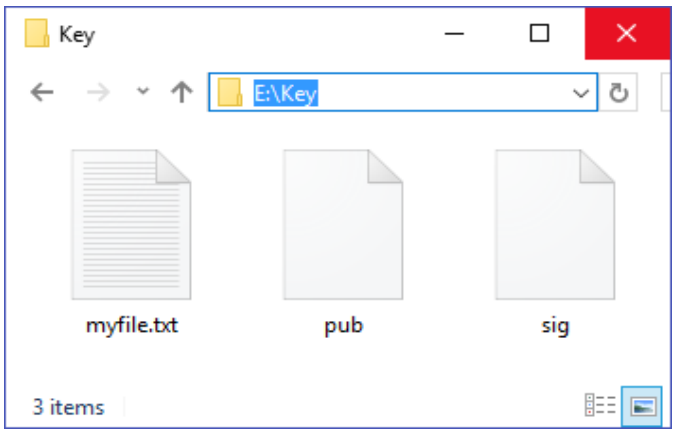
Write a program to digitally sign MIME to create an 'opaque' signature.

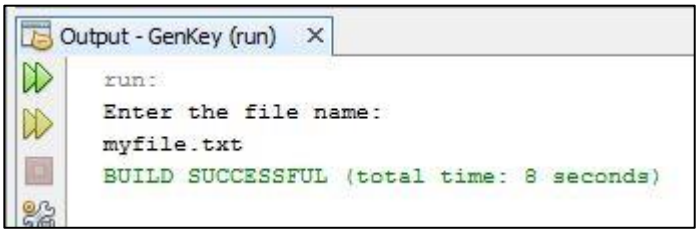
Code:

Genkey.java

```
import java.io.*;
import java.security.*;
public class GenKey {
   public static void main(String[] args) {
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      System.out.println("Enter the file name:");
      try {
         String f_name = br.readLine();
         KeyPairGenerator kpg = KeyPairGenerator.getInstance("DSA");
         SecureRandom rdm = SecureRandom.getInstance("SHA1PRNG");
         kpg.initialize(1024, rdm);
         KeyPair pair = kpg.genKeyPair();
         PrivateKey pr = pair.getPrivate();
         PublicKey pub = pair.getPublic();
         Signature dsa = Signature.getInstance("SHA1withDSA");
         dsa.initSign(pr);
         FileInputStream fs = new FileInputStream("E:\\Key\\" + f_name);
         BufferedInputStream bufin = new BufferedInputStream(fs);
         byte[] buffer = new byte[1024];
         int len;
         while (bufin.available() != 0) {
            len = bufin.read(buffer);
            dsa.update(buffer, 0, len);
         }
         bufin.close();
         byte[] realSig = dsa.sign();
         FileOutputStream sigfos = new FileOutputStream("E:\\Key\\sig");
         sigfos.write(realSig);
         sigfos.close();
         byte[] key = pub.getEncoded();
         FileOutputStream keyfos = new FileOutputStream("E:\\Key\\pub");
         keyfos.write(key);
         keyfos.close();
      } catch (Exception e) {
         e.printStackTrace();
      }}}
```

Output:



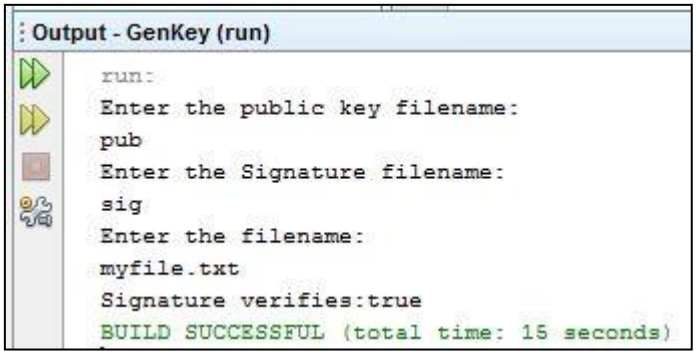


vfKey.java

```
import java.io.*;
import java.security.*;
import java.security.spec.X509EncodedKeySpec;
class vfKey {
  public static void main(String[] args) {
    try
    {BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      System.out.println("Enter the public key filename:");
      String pub = br.readLine();
      System.out.println("Enter the Signature filename:");
      String sigfile = br.readLine();
      System.out.println("Enter the filename:");
      String f_name = br.readLine();
      FileInputStream keyfis = new FileInputStream("E:\\Key\\" + pub);
      byte[] encKey = new byte[keyfis.available()];
      keyfis.read(encKey);
      keyfis.close();
      X509EncodedKeySpec pubKeySpec = new X509EncodedKeySpec(encKey);
      KeyFactory keyFactory = KeyFactory.getInstance("DSA");
      PublicKey pubKey = keyFactory.generatePublic(pubKeySpec);
      FileInputStream sigfis = new FileInputStream("E:\\Key\\" + sigfile);
      byte[] sigToVerify = new byte[sigfis.available()];
```

```
        sigfis.read(sigToVerify);
        sigfis.close();
        Signature sig = Signature.getInstance("SHA1withDSA");
        sig.initVerify(pubKey);
        FileInputStream datafis = new FileInputStream("E:\\Key\\" + f_name);
        BufferedInputStream bufin = new BufferedInputStream(datafis);
        byte[] buffer = new byte[1024];
        int len;
        while(bufin.available()!=0) {
            len = bufin.read(buffer);
            sig.update(buffer, 0, len);
        } bufin.close();
        boolean verifies = sig.verify(sigToVerify);
        System.out.println("Signature verifies:" +verifies);
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }}}
```

Output:



Practical 7

Write a program to generate DSA SSH key.

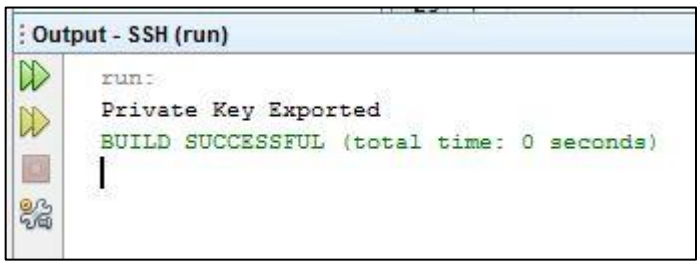
Code:

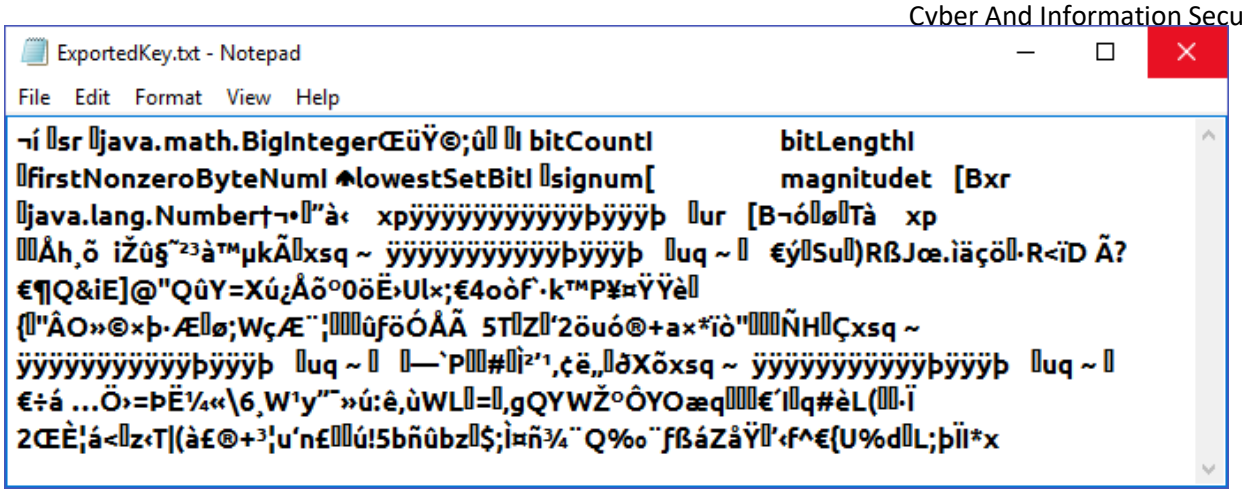
SSHExport.java

```
import java.io.*;
import java.security.*;
import java.security.spec.DSAPrivateKeySpec;

public class SSHExport {
   public static void main(String[] args) {
      try
      {
         KeyPairGenerator kpg = KeyPairGenerator.getInstance("DSA");
         SecureRandom rnd = SecureRandom.getInstance("SHA1PRNG","SUN");
         kpg.initialize(1024,rnd);
         KeyPair kp = kpg.generateKeyPair();
         Class spec = Class.forName("java.security.spec.DSAPrivateKeySpec");
         KeyFactory kf = KeyFactory.getInstance("DSA");
         DSAPrivateKeySpec ks = (DSAPrivateKeySpec)kf.getKeySpec(kp.getPrivate(), spec);
         FileOutputStream fos = new FileOutputStream("ExportedKey.txt");
         ObjectOutputStream oos = new ObjectOutputStream(fos);
         oos.writeObject(ks.getX());
         oos.writeObject(ks.getP());
         oos.writeObject(ks.getQ());
         oos.writeObject(ks.getG());
         System.out.println("Private Key Exported");
      }
      catch(Exception e)
      {
         e.printStackTrace();
      }}}
```

Output:





SSHImport.java

```
import java.io.*;
import java.math.BigInteger;
import java.security.*;
import java.security.spec.DSAPrivateKeySpec;
public class SSHImport {
   public static void main(String[] args) {
      try {
         FileInputStream fis = new FileInputStream("exportedKey.txt");
         ObjectInputStream ois = new ObjectInputStream(fis);
         DSAPrivateKeySpec ks = new DSAPrivateKeySpec((BigInteger) ois.readObject(), (BigInteger)
ois.readObject(), (BigInteger) ois.readObject(), (BigInteger) ois.readObject());
         KeyFactory kf = KeyFactory.getInstance("DSA");
         PrivateKey pk = kf.generatePrivate(ks);
         System.out.println("Got private key.");
      } catch (FileNotFoundException e) {
         System.out.println("Key not found.");
      } catch (Exception e) {
         System.out.println("Key is corrupted.");
      }}}
```

Output: