```
//ASS1_Set_A_a

import java.util.*;
class Array_Linked
{
  public static void main(String args[])
  {
    ArrayList<String> Ar = new ArrayList<String>();
    System.out.println("Initial Size Of ArrayList : " + Ar.size());

    Scanner sc = new Scanner(System.in);
    System.out.println("How many Cities : ");
    int n = sc.nextInt();
    for(int i=0;i<n;i++)
    {
     System.out.println("Enter City : ");
     String S = sc.next();
     Ar.add(S);
    }
    System.out.println("All city Names : " + Ar);
    Ar.clear();
   System.out.println("All Elements Are removed: " + Ar);
  }
}


//ASS1_Set_A_b

import java.util.*;
class Linked
{
  public static void main(String args[])
  {
   LinkedList<String> Ar = new LinkedList<String>();
    System.out.println("Initial Size Of LinkedList : " + Ar.size());

    Scanner sc = new Scanner(System.in);
    System.out.println("How many Friends : ");
    int n = sc.nextInt();
    for(int i=0;i<n;i++)
    {
     System.out.println("Enter Friends : ");
     String S = sc.next();
     Ar.add(S);
    }
    System.out.println("All Friends Names: " + Ar);
  }
}



//ASS1_Set_A_c

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

```
{
  public static void main(String args[])
  {
    Set<String> ts = new TreeSet<>();

      ts.add("Red");
      ts.add("Blue");
      ts.add("Green");

      System.out.println("All Colors :"+ts);
  }
}



//ASS1_Set_A_d

import java.util.*;
class Hash_Table
{
  public static void main(String args[])
  {
      Hashtable<Integer, String> ht1 = new Hashtable<>();

      ht1.put(11112232, "Sk");
      ht1.put(2223321, "TS");
      ht1.put(3333234, "VS");

      System.out.println("Students Mobile Numbers & Names : " + ht1);
   }
}


//ASS1_Set_B_a

import java.util.*;
class Tree_Set
{
  public static void main(String args[])
  {
      TreeSet<Object> ints = new TreeSet<Object>();
      ints.add(2);
      ints.add(20);
      ints.add(10);
      ints.add(7);
      ints.add(7);
      ints.add(3);

      TreeSet<Object> intsReverse = (TreeSet<Object>)ints.descendingSet();

      System.out.println("Acending Order: "+ints);
      System.out.println("Decending Order: "+intsReverse);
   }
}
```

```
//ASS1_Set_B_b

import java.util.*;
class Tree_Set1
{
   static Map<String, Integer> map = new HashMap<>();
   public static void sortbykey()
   {

    TreeMap<String, Integer> sorted = new TreeMap<>();
    sorted.putAll(map);
       for (Map.Entry<String, Integer> entry : sorted.entrySet())
           System.out.println("Key = " + entry.getKey() +", Value = " + entry.getValue());
   }
       public static void main(String args[])
   {

       map.put("Jayant", 80);
       map.put("Abhishek", 90);
       map.put("Anushka", 80);
       map.put("Amit", 75);
       map.put("Danish", 40);

       sortbykey();
   }
}


//ASS1_Set_B_c

import java.io.*;
import java.util.*;
class Phonebook
{
public static void main(String args[])
{
try
{
FileInputStream fis=new FileInputStream("G:/DCIM/Satish/JAVA/ASSIGNMENTS/Myfile.txt");
Scanner sc=new Scanner(fis).useDelimiter("\t");
Hashtable<String,String> ht=new Hashtable<String,String> ();
String[] strarray;
String a,str;
while(sc.hasNext())
{
a=sc.nextLine();
strarray=a.split("\t");
ht.put(strarray[0],strarray[1]);
System.out.println("hash table values are : "+strarray[0]+":"+strarray[1]);
}
Scanner s=new Scanner(System.in);
System.out.println("Enter the name as given in the phone book");
str=s.next();
if(ht.containsKey(str))
```

```
{
System.out.println("phone no is"+ht.get(str));
}
else
{
System.out.println("Name is not matched");
}
}
catch(Exception e)
{
System.out.println(e);
}
}
}



//ASS1_Set_C_a

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.*;


class SS extends JFrame implements ActionListener
{
	JTextField t1,t2,t3;
	JButton b1,b2,b3;
	JTextArea t;
	JPanel p1,p2;

	Hashtable ts;
	Slip16_2()
	{
		ts=new Hashtable();
		t1=new JTextField(10);
		t2=new JTextField(10);
		t3=new JTextField(10);

		b1=new JButton("Add");
		b2=new JButton("Search");
		b3=new JButton("Remove");

		t=new JTextArea(20,20);
		p1=new JPanel();
		p1.add(t);

		p2= new JPanel();
		p2.setLayout(new GridLayout(2,3));
		p2.add(t1);
		p2.add(t2);
		p2.add(b1);
		p2.add(t3);
		p2.add(b2);
```

```
        p2.add(b3);

        add(p1);
        add(p2);

        b1.addActionListener(this);
        b2.addActionListener(this);
        b3.addActionListener(this);

        setLayout(new FlowLayout());
        setSize(500,500);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


}
public void actionPerformed(ActionEvent e)
{
        if(b1==e.getSource())
        {
                String name = t1.getText();
                int code = Integer.parseInt(t2.getText());
                ts.put(name,code);
                Enumeration k=ts.keys();
                Enumeration v=ts.elements();
                String msg="";
                while(k.hasMoreElements())
                {
                        msg=msg+k.nextElement()+" = "+v.nextElement()+"\n";
                }
                t.setText(msg);
                t1.setText("");
                t2.setText("");
        }
        else if(b2==e.getSource())
        {
                String name = t3.getText();

                if(ts.containsKey(name))
                {
                        t.setText(ts.get(name).toString());
                }

                else
                        JOptionPane.showMessageDialog(null,"City not found ...");
        }
        else if(b3==e.getSource())
        {
                String name = t3.getText();

                if(ts.containsKey(name))
                {
                        ts.remove(name);
                        JOptionPane.showMessageDialog(null,"City Deleted ...");
                }
```

```
                        else
                                JOptionPane.showMessageDialog(null,"City not found ...");
                }
        }
        public static void main(String a[])
        {
                new SS();
        }
}



//ASS1_Set_C_b


import java.util.Scanner;

class Node
{
   protected int data;
   protected Node link;


   public Node()
   {
      link = null;
      data = 0;
   }

   public Node(int d,Node n)
   {
      data = d;
      link = n;
   }

   public void setLink(Node n)
   {
      link = n;
   }

   public void setData(int d)
   {
      data = d;
   }

   public Node getLink()
   {
      return link;
   }

   public int getData()
   {
      return data;
   }
```

```
}


class linkedList
{
   protected Node start;
   protected Node end ;
   public int size ;


   public linkedList()
   {
      start = null;
      end = null;
      size = 0;
   }

   public boolean isEmpty()
   {
      return start == null;
   }

   public int getSize()
   {
      return size;
   }

   public void insertAtStart(int val)
   {
      Node nptr = new Node(val, null);
      size++ ;
      if(start == null)
      {
         start = nptr;
         end = start;
      }
      else
      {
         nptr.setLink(start);
         start = nptr;
      }
   }

   public void insertAtEnd(int val)
   {
      Node nptr = new Node(val,null);
      size++ ;
      if(start == null)
      {
         start = nptr;
         end = start;
      }
      else
      {
         end.setLink(nptr);
```

```
        end = nptr;
    }
}

public void insertAtPos(int val , int pos)
{
    Node nptr = new Node(val, null);
    Node ptr = start;
    pos = pos - 1 ;
    for (int i = 1; i < size; i++)
    {
        if (i == pos)
        {
            Node tmp = ptr.getLink() ;
            ptr.setLink(nptr);
            nptr.setLink(tmp);
            break;
        }
        ptr = ptr.getLink();
    }
    size++ ;
}

public void deleteAtPos(int pos)
{
    if (pos == 1)
    {
        start = start.getLink();
        size--;
        return ;
    }
    if (pos == size)
    {
        Node s = start;
        Node t = start;
        while (s != end)
        {
            t = s;
            s = s.getLink();
        }
        end = t;
        end.setLink(null);
        size --;
        return;
    }
    Node ptr = start;
    pos = pos - 1 ;
    for (int i = 1; i < size - 1; i++)
    {
        if (i == pos)
        {
            Node tmp = ptr.getLink();
            tmp = tmp.getLink();
            ptr.setLink(tmp);
            break;
```

```
            }
            ptr = ptr.getLink();
        }
        size-- ;
    }

    public void display()
    {
        System.out.print("\nSingly Linked List = ");
        if (size == 0)
        {
            System.out.print("empty\n");
            return;
        }
        if (start.getLink() == null)
        {
            System.out.println(start.getData() );
            return;
        }
        Node ptr = start;
        System.out.print(start.getData()+ "->");
        ptr = start.getLink();
        while (ptr.getLink() != null)
        {
            System.out.print(ptr.getData()+ "->");
            ptr = ptr.getLink();
        }
        System.out.print(ptr.getData()+ "\n");
    }
}


class SinglyLinkedList
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);

        linkedList list = new linkedList();
        System.out.println("Singly Linked List Test\n");
        char ch;

        do
        {
            System.out.println("\nSingly Linked List Operations\n");
            System.out.println("1. insert at begining");
            System.out.println("2. insert at end");
            System.out.println("3. insert at position");
            System.out.println("4. delete at position");
            System.out.println("5. check empty");
            System.out.println("6. get size");
            int choice = scan.nextInt();
            switch (choice)
            {
            case 1 :
```

```
            System.out.println("Enter integer element to insert");
            list.insertAtStart( scan.nextInt() );
            break;
         case 2 :
            System.out.println("Enter integer element to insert");
            list.insertAtEnd( scan.nextInt() );
            break;
         case 3 :
            System.out.println("Enter integer element to insert");
            int num = scan.nextInt() ;
            System.out.println("Enter position");
            int pos = scan.nextInt() ;
            if (pos <= 1 || pos > list.getSize() )
               System.out.println("Invalid position\n");
            else
               list.insertAtPos(num, pos);
            break;
         case 4 :
            System.out.println("Enter position");
            int p = scan.nextInt() ;
            if (p < 1 || p > list.getSize() )
               System.out.println("Invalid position\n");
            else
               list.deleteAtPos(p);
            break;
         case 5 :
            System.out.println("Empty status = "+ list.isEmpty());
            break;
         case 6 :
            System.out.println("Size = "+ list.getSize() +" \n");
            break;
          default :
            System.out.println("Wrong Entry \n ");
            break;
         }

         list.display();
         System.out.println("\nDo you want to continue (Type y or n) \n");
         ch = scan.next().charAt(0);
      } while (ch == 'Y'|| ch == 'y');
   }
}



//ASS2_Set_A_a


import java.lang.Thread;
class ThreadA extends Thread
{
  public void run()
  {
   for(int i=1;i<=10;i++)
     System.out.println("COVID19");
```

```
System.out.println("Exiting From Thread A");
  }

}


class ThreadB extends Thread
{
   public void run()
   {
    for(int j=1;j<=20;j++)
      System.out.println("LOCKDOWN2020");
System.out.println("Exiting From Thread B");
  }
}

class ThreadC extends Thread
{
   public void run()
   {
    for(int k=1;k<=30;k++)
      System.out.println("VACCINATED2021");
System.out.println("Exiting From Thread C");
  }

}


class SK
{
  public static void main(String args[])
  {
   ThreadA a = new ThreadA();
   ThreadB b = new ThreadB();
   ThreadC c = new ThreadC();

   a.start();
   b.start();
   c.start();

   System.out.println("MultiThreading Is Over....");
  }
}



//ASS2_Set_A_b


import java.io.*;
import java.lang.Thread;

class GFG {
    public static void main(String[] args)
    {
```

```
        try {
            for (int i = 100; i >=1; i--)
            {

                Thread.sleep(6);
                System.out.println(i);
            }
        }
        catch (Exception e) {

            System.out.println(e);
        }
    }
}




//ASS2_Set_A_c


import java.util.LinkedList;
import java.lang.Thread;
class Threadexample {
    public static void main(String[] args)
        throws InterruptedException
    {
        // Object of a class that has both produce()
        // and consume() methods
        final PC pc = new PC();

        // Create producer thread
        Thread t1 = new Thread(new Runnable() {
            @Override
            public void run()
            {
                try {
                    pc.produce();
                }
                catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });

        // Create consumer thread
        Thread t2 = new Thread(new Runnable() {
            @Override
            public void run()
            {
                try {
                    pc.consume();
                }
                catch (InterruptedException e) {
                    e.printStackTrace();
```

```
            }
        }
    });

    // Start both threads
    t1.start();
    t2.start();

    // t1 finishes before t2
    t1.join();
    t2.join();
}

// This class has a list, producer (adds items to list
// and consumer (removes items).
public static class PC {

    // Create a list shared by producer and consumer
    // Size of list is 2.
    LinkedList<Integer> list = new LinkedList<>();
    int capacity = 2;

    // Function called by producer thread
    public void produce() throws InterruptedException
    {
        int value = 0;
        while (true) {
            synchronized (this)
            {
                // producer thread waits while list
                // is full
                while (list.size() == capacity)
                    wait();

                System.out.println("Producer produced-"
                                + value);

                // to insert the jobs in the list
                list.add(value++);

                // notifies the consumer thread that
                // now it can start consuming
                notify();

                // makes the working of program easier
                // to understand
                Thread.sleep(1000);
            }
        }
    }

    // Function called by consumer thread
    public void consume() throws InterruptedException
    {
        while (true) {
```

```
            synchronized (this)
            {
               // consumer thread waits while list
               // is empty
               while (list.size() == 0)
                  wait();

               // to retrieve the first job in the list
               int val = list.removeFirst();

               System.out.println("Consumer consumed-"
                                + val);

               // Wake up producer thread
               notify();

               // and sleep
               Thread.sleep(1000);
            }
         }
      }
   }
}



//ASS2_Set_B_a


 import java.util.*;
 import java.lang.Thread;
class thread implements Runnable
{
        Thread t;
                int i,no,sum;
                int a[]=new int[1000];
                thread(String s,int n)
                {
                        Random rs = new Random();
                                t=new Thread(this,s);
                                no=n;
                                int j=0;
                                for(i=1;i<=1000;i++)
                                {
                                        a[j]=rs.nextInt()%100;;
                                                j++;
                                }
                        t.start();
                }
        public void run() {
                for(i=0;i<100;i++)
                {
                        sum=sum+a[no];
                                no++;
                }
```

```
                System.out.println("Sum = "+sum);
                System.out.println("Avg ="+sum/100);
        }

}
class SSK
{
        public static void main(String[] arg) throws InterruptedException
        {
                thread t1=new thread("g",1);
                        t1.t.join();
                        thread t2=new thread("r",100);
                        t2.t.join();
                        thread t3=new thread("s",200);
                        t3.t.join();
                        thread t4=new thread("t",300);
                        t4.t.join();
                        thread t5=new thread("p",400);
                        t5.t.join();
                thread t6=new thread("p",500);
                        t5.t.join();
                thread t7=new thread("p",600);
                        t5.t.join();
                thread t8=new thread("p",700);
                        t5.t.join();
                thread t9=new thread("p",800);
                        t5.t.join();
                thread t10=new thread("p",900);
                        t5.t.join();

        }
}



//ASS2_Set_B_b

import java.io.*;
import java.lang.Thread;
class SearchThread extends Thread
{
   File f1;
   String fname;
   static String str;
   String line;
    LineNumberReader reader = null;
   SearchThread(String fname)
   {
      this.fname=fname;
      f1=new File(fname);
   }
   public void run()
   {
      try
      {
```

```
        FileReader fr=new FileReader(f1);
        reader=new  LineNumberReader(fr);
        while((line=reader.readLine())!=null)
        {
           if(line.indexOf(str)!=-1)
           {
              System.out.println("string found in "+fname+"at "+reader.getLineNumber()+"line");
              stop();
           }
        }
     }
     catch(Exception e)
     {
     }
  }
public static void main(String[] args) throws IOException
{
     Thread t[]=new Thread[20];
     BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
     System.out.println("Enter String to search");
     str=br.readLine();

     FilenameFilter filter = new FilenameFilter()
     {
        public boolean accept(File file, String name)
     {
           if (name.endsWith(".txt"))
        {
              return true;
           }
        else
           {
            return false;
           }
        }
};

File dir1 = new File(".");
File[] files = dir1.listFiles(filter);
   if (files.length == 0)
{
   System.out.println("no files available with this extension");
}
else
   {
       for(int i=0;i<files.length;i++)
       {
          for (File aFile : files)
             {
               t[i]=new SearchThread(aFile.getName());
               t[i].start();
                }
       }
   }
}
```

```
}


//ASS2_Set_B_c


import java.util.Random;
import java.lang.Thread;
class Square extends Thread

{

 int x;

 Square(int n)

 {

 x = n;

 }

 public void run()

 {

 int sqr = x * x;

 System.out.println("Square of " + x + " = " + sqr );

 }

}

class Cube extends Thread

{

 int x;

 Cube(int n)

 {x = n;

 }

 public void run()

 {

 int cub = x * x * x;

 System.out.println("Cube of " + x + " = " + cub );
```

```
 }
}
class Number extends Thread
{
 public void run()
 {
 Random random = new Random();
 for(int i =0; i<5; i++)
 {
 int randomInteger = random.nextInt(100);
 System.out.println("Random Integer generated : " + randomInteger);
 Square s = new Square(randomInteger);
 s.start();
 Cube c = new Cube(randomInteger);
 c.start();
 try {
 Thread.sleep(1000);
} catch (InterruptedException ex) {
 System.out.println(ex);
}
 }
 }
}
class Thr {
 public static void main(String args[])
 {
 Number n = new Number();
 n.start();
```

```
 }

}



//ASS2_Set_C_a

import javax.swing.*;
import javax.swing.event.*;
import java.awt.*;
import java.awt.event.*;
class TrafficLightSimulator extends JFrame implements ItemListener {
    JLabel lbl1, lbl2;
    JPanel nPanel, cPanel;
    CheckboxGroup cbg;
    public TrafficLightSimulator() {
        setTitle("Traffic Light Simulator");
        setSize(600,400);
        setLayout(new GridLayout(2, 1));
        nPanel = new JPanel(new FlowLayout());
        cPanel = new JPanel(new FlowLayout());
        lbl1 = new JLabel();
        Font font = new Font("Verdana", Font.BOLD, 70);
        lbl1.setFont(font);
        nPanel.add(lbl1);
        add(nPanel);
        Font fontR = new Font("Verdana", Font.BOLD, 20);
        lbl2 = new JLabel("Select Lights");
        lbl2.setFont(fontR);
        cPanel.add(lbl2);
        cbg = new CheckboxGroup();
        Checkbox rbn1 = new Checkbox("Red Light", cbg, false);
        rbn1.setBackground(Color.RED);
        rbn1.setFont(fontR);
        cPanel.add(rbn1);
        rbn1.addItemListener(this);
        Checkbox rbn2 = new Checkbox("Orange Light", cbg, false);
        rbn2.setBackground(Color.ORANGE);
        rbn2.setFont(fontR);
        cPanel.add(rbn2);
        rbn2.addItemListener(this);
        Checkbox rbn3 = new Checkbox("Green Light", cbg, false);
        rbn3.setBackground(Color.GREEN);
        rbn3.setFont(fontR);
        cPanel.add(rbn3);
        rbn3.addItemListener(this);
        add(cPanel);
        setVisible(true);
        // to close the main window
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
    // To read selected item
    public void itemStateChanged(ItemEvent i) {
        Checkbox chk = cbg.getSelectedCheckbox();
```

```
        String str=chk.getLabel();
        char choice=str.charAt(0);
        switch (choice) {
        case 'R':lbl1.setText("STOP");
                 lbl1.setForeground(Color.RED);
                 break;
        case 'O':lbl1.setText("READY");
                 lbl1.setForeground(Color.ORANGE);
                 break;
        case 'G':lbl1.setText("GO");
                 lbl1.setForeground(Color.GREEN);
                 break;
        }
    }
    // main method
    public static void main(String[] args) {
        new TrafficLightSimulator();
    }
}




//ASS2_Set_C_b


import java.awt.*;
import java.util.Formatter;
import javax.swing.*;
import java.lang.Thread;

class BouncingBallSimple extends JPanel {

   private static final int BOX_WIDTH = 640;
   private static final int BOX_HEIGHT = 480;


   private float ballRadius = 200;
   private float ballX = ballRadius + 50;
   private float ballY = ballRadius + 20;
   private float ballSpeedX = 3;
   private float ballSpeedY = 2;

   private static final int UPDATE_RATE = 30;
   public BouncingBallSimple() {
      this.setPreferredSize(new Dimension(BOX_WIDTH, BOX_HEIGHT));


      Thread gameThread = new Thread() {
         public void run() {
            while (true) {
               ballX += ballSpeedX;
               ballY += ballSpeedY;

               if (ballX - ballRadius < 0) {
```

```
            ballSpeedX = -ballSpeedX;
            ballX = ballRadius;
          } else if (ballX + ballRadius > BOX_WIDTH) {
            ballSpeedX = -ballSpeedX;
            ballX = BOX_WIDTH - ballRadius;
          }

          if (ballY - ballRadius < 0) {
            ballSpeedY = -ballSpeedY;
            ballY = ballRadius;
          } else if (ballY + ballRadius > BOX_HEIGHT) {
            ballSpeedY = -ballSpeedY;
            ballY = BOX_HEIGHT - ballRadius;
          }

          repaint();
          try {
            Thread.sleep(1000 / UPDATE_RATE);
          } catch (InterruptedException ex) { }
        }
      }
    };
    gameThread.start();
  }


  @Override
  public void paintComponent(Graphics g) {
    super.paintComponent(g);

    g.setColor(Color.BLACK);
    g.fillRect(0, 0, BOX_WIDTH, BOX_HEIGHT);


    g.setColor(Color.ORANGE);
    g.fillOval((int) (ballX - ballRadius), (int) (ballY - ballRadius),
        (int)(2 * ballRadius), (int)(2 * ballRadius));


    g.setColor(Color.ORANGE);
    g.setFont(new Font("Courier New", Font.PLAIN, 12));
    StringBuilder sb = new StringBuilder();
    Formatter formatter = new Formatter(sb);
    formatter.format("Ball @(%3.0f,%3.0f) Speed=(%2.0f,%2.0f)", ballX, ballY,
        ballSpeedX, ballSpeedY);
    g.drawString(sb.toString(), 20, 30);
  }


  public static void main(String[] args) {

    javax.swing.SwingUtilities.invokeLater(new Runnable() {
      public void run() {

        JFrame frame = new JFrame("A Bouncing Ball");
```

```
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setContentPane(new BouncingBallSimple());
            frame.pack();
            frame.setVisible(true);
         }
      });
   }
}
```

```
//ASS2_Set_C_c


import java.io.*;
import java.util.*;
import java.lang.Thread;

class Sender
{
   public void send(String msg)
   {
      System.out.println("\t"  + msg );
      try
      {
         Thread.sleep(1000);
      }
      catch (Exception e)
      {
         System.out.println("Thread  interrupted.");
      }
      System.out.println("\n" + msg);
   }
}


class ThreadedSend extends Thread
{
   private String msg;
   Sender  sender;


   ThreadedSend(String m,  Sender obj)
   {
      msg = m;
      sender = obj;
   }

   public void run()
   {

      synchronized(sender)
      {
```

```
            sender.send(msg);
        }
    }
}


class SyncDemo
{
    public static void main(String args[])
    {
        Sender send = new Sender();
        ThreadedSend S1 =
            new ThreadedSend( " Hi " , send );
        ThreadedSend S2 =
            new ThreadedSend( " Good Bye Corona \n" , send );


        S1.start();
        S2.start();


        try
        {
            S1.join();
            S2.join();
        }
        catch(Exception e)
        {
            System.out.println("Interrupted");
        }
    }
}



//ASS3_Set_A_a


import java.sql.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.*;


class Project extends JFrame implements ActionListener
{
            JLabel l1,l2,l3;
            JTextField t1,t2,t3;
            JButton b1,b2,b3;
            String sql;
            JPanel p,p1;
            Connection con;
            PreparedStatement ps;
```

```
JTable t;
JScrollPane js;
Statement stmt ;
ResultSet rs ;
ResultSetMetaData rsmd ;
int columns;
Vector columnNames = new Vector();
Vector data = new Vector();

Slip13_2()
{

        l1 = new JLabel("Enter no :");
        l2 = new JLabel("Enter name :");
        l3 = new JLabel("percentage :");

        t1 = new JTextField(20);
        t2 = new JTextField(20);
        t3 = new JTextField(20);

        b1 = new JButton("Save");
        b2 = new JButton("Display");
        b3 = new JButton("Clear");

        b1.addActionListener(this);
        b2.addActionListener(this);
        b3.addActionListener(this);

        p=new JPanel();
        p1=new JPanel();
        p.add(l1);
        p.add(t1);
        p.add(l2);
        p.add(t2);
        p.add(l3);
        p.add(t3);

        p.add(b1);
        p.add(b2);
        p.add(b3);

        add(p);
        setLayout(new GridLayout(2,1));
        setSize(600,800);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


}

public void actionPerformed(ActionEvent e)
{
        if((JButton)b1==e.getSource())
        {
```

```
                        int no = Integer.parseInt(t1.getText());
                        String name = t2.getText();
                        int p = Integer.parseInt(t3.getText());
                        System.out.println("Accept Values");
                        try
                        {
                                Class.forName(“org.postgresql.Driver”);
con=DriverManager.getConnection(“jdbc:postgresql://192.168.100.254/Bill”,”oracle”,”oracle”);

sql = "insert into stud values(?,?,?)";
                                ps = con.prepareStatement(sql);
                                ps.setInt(1,no);
                                ps.setString(2, name);
                                ps.setInt(3,p);
                                System.out.println("values set");
                                int n=ps.executeUpdate();
                                if(n!=0)
                                {
                                        JOptionPane.showMessageDialog(null,"Record insered ...");

                                }

                                else
                                        JOptionPane.showMessageDialog(null,"Record NOT inserted ");

                        }//end of try
                        catch(Exception ex)
                        {
                                System.out.println(ex);
                                //ex.printStackTrace();
                        }

                }//end of if
                else if((JButton)b2==e.getSource())
                {
                        try
                        {
                                Class.forName(“org.postgresql.Driver”);
con=DriverManager.getConnection(“jdbc:postgresql://192.168.100.254/Bill”,”oracle”,”oracle”);
                                System.out.println("Connected");
                                stmt=con.createStatement();
                                rs = stmt.executeQuery("select * from stud");
                                rsmd = rs.getMetaData();
                                columns = rsmd.getColumnCount();

                                //Get Columns name
                                for(int i = 1; i <= columns; i++)
                                {
                                        columnNames.addElement(rsmd.getColumnName(i));
                                }

                                //Get row data
                                while(rs.next())
                                {
                                        Vector row = new Vector(columns);
```

```
                        for(int i = 1; i <= columns; i++)
                        {
                                row.addElement(rs.getObject(i));
                        }
                        data.addElement(row);
                }

                t = new JTable(data, columnNames);
                js = new JScrollPane(t);

                p1.add(js);
                add(p1);

                setSize(600, 600);
                setVisible(true);
            }
            catch(Exception e1)
            {
                System.out.println(e1);
            }
        }
        else
        {
            t1.setText(" ");
            t2.setText(" ");
            t3.setText(" ");

        }
    }//end of method

    public static void main(String a[])
    {
        Project ob = new Project();
    }
}
```