//ASS1_Set_A_a

- Write a java program to accept names of 'n' cities, insert same into array list
collection and display the contents of same array list, also remove all these elements.

```
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

- Write a java program to read 'n' names of your friends, store it into linked list, also display contents of the same.

```
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

- Write a program to create a new tree set, add some colors (string) and print out the tree set.

```
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

- Create the hash table that will maintain the mobile number and student name. Display the contact list.

```
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

- Accept 'n' integers from the user. Store and display integers in sorted order having proper collection class. The collection should not accept duplicate elements.

```
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

- Write a program to sort HashMap by keys and display the details before sorting and after sorting.

```
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

- Write a program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t).it takes a name or phone number as input and prints the corresponding other value from the hash table (hint: use hash tables)

```
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

a) Create a java application to store city names and their STD codes using appropriate collection. The GUI should allow the following operations:

i. Add a new city and its code (No duplicates)
ii. Remove a city from the collection
3.Search for a city name and display the code

```
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
Write a program to create link list of integer objects. Do the following:
i. add element at first position
ii. delete last element display the size of link list
3.display the size of linked list

```
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");
    }
}
```

- Write a program to create link list of integer objects. Do the following:

i. add element at first position
ii. delete last element
3.display the size of link list

```
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

a) Program to define a thread for printing text on output screen for 'n' number of times. Create 3 threads and run them. Pass the text 'n' parameters to the thread constructor. Example:

i. First thread prints "COVID19" 10 times.

ii. Second thread prints "LOCKDOWN2020" 20 times iii. Third thread prints "VACCINATED2021" 30 times in

```
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
```

b) Write a program in which thread sleep for 6 sec in the loop in reverse order from 100

to 1 and change the name of thread.

```
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
```

c) Write a program to solve producer consumer problem in which a producer produces a value and consumer consume the value before producer generate the next value. (Hint: use thread synchronization)

```
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

a) Write a program to calculate the sum and average of an array of 1000 integers (generated randomly) using 10 threads. Each thread calculates the sum of 100 integers. Use these values to calculate average. [Use join method].

```
 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
```

b) Write a program for a simple search engine. Accept a string to be searched. Search for the string in all text files in the current folder. Use a separate thread for each file. The result should display the filename, line number where the string is found.

```
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
```

c) Write a program that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.

```
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

a) Write a program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with "stop" or "ready" or "go"should appear above the buttons in a selected color. Initially there is no message shown.

```
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

b) Write a program to create a thread for moving a ball inside a panel vertically. The ball should be created when the user clicks on the start button.

```
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
```

c) Using the concepts of thread synchronization create two threads as sender and receiver. Sender thread will set a message to the receiver thread that will display the message on console. The sender thread accepts the input message from console. Continue this process until sender sets the message as "Good Bye Corona".

```
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
```

a) Create a PROJECT table with fields project_id, Project_name, Project_description, Project Status. etc. Insert values in the table. Display all the details of the PROJECT table in a tabular format on the screen.(using swing).

```
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();
        }
}
```

b) Write a program to display information about the database and list all the tables in the database. (Use DatabaseMetaData).

```
import java.sql.*;
import java.io.*;
public class DBMetaData
{
  public static void main(String[] args) throws Exception
  {
     ResultSet rs = null;
      Class.forName("org.postgresql.Driver");
       Connection conn =
DriverManager.getConnection("jdbc:postgresql://localhost/dbtry","postgres","redhat")
;
    DatabaseMetaData dbmd = conn.getMetaData();
    System.out.println("Database Product name = " + dbmd.getDatabaseProductName());
    System.out.println("User name = " + dbmd.getUserName());
    System.out.println("Database driver  name= " + dbmd.getDriverName());
    System.out.println("Database driver version = "+ dbmd.getDriverVersion());
    System.out.println("Database product name = " + dbmd.getDatabaseProductName());
    System.out.println("Database Version = " + dbmd.getDriverMajorVersion());
    rs = dbmd.getTables(null,null,null, new String[]{"TABLE"});
    System.out.println("List of tables...");
    while(rs.next())
    {
          String tblName = rs.getString("TABLE_NAME");
          System.out.println("Table : "+ tblName);
    }
    conn.close();
  }
}
```

c) Write a program to display information about all columns in the DONAR table using

ResultSetMetaData.

```
import java.sql.*;
import java.io.*;
public class ResultSetMetaData
{
  public static void main(String[] args) throws Exception
  {



    Statement stmt;
     Class.forName("org.postgresql.Driver");
       Connection conn =
DriverManager.getConnection("jdbc:postgresql://localhost/stud","postgres","password"
);
    stmt = conn.createStatement();
   ResultSet rs = stmt.executeQuery("Select * from student");
    java.sql.ResultSetMetaData rsmd = rs.getMetaData();
    int noOfColumns = rsmd.getColumnCount();
    System.out.println("Number of columns = " + noOfColumns);
    for(int i=1; i<=noOfColumns; i++)
    {
       System.out.println("Column No : " + i);
       System.out.println("Column Name : " + rsmd.getColumnName(i));
       System.out.println("Column Type : " + rsmd.getColumnTypeName(i));
      System.out.println("Column display size : " + rsmd.getColumnDisplaySize(i));
    }
    conn.close();
  }
}
```

SET B .

a) Create a MOBILE table with fields Model Number, Model_Name, Model_Color, Sim Type, NetworkType, BatteryCapacity, InternalStorage, RAM and Processor Type. Insert values in the table. Write a menu driven program to pass the input using Command line argument to perform the following operations on MOBILE table.

1. Insert 2. Modify 3. Delete 4. Search 5. View All 6. Exit

```
import java.sql.*;
import java.io.*;
class JDBCMenu
{
  public static void main(String[] args) throws Exception
  {

    Statement stmt =  null;
    ResultSet rs = null;
    PreparedStatement ps1 = null, ps2=null;
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    String name;
    int r,choice;
    float per;
    Class.forName("org.postgresql.Driver");
    Connection conn = 
DriverManager.getConnection("jdbc:postgresql://localhost/stud","postgres","password"
);
    stmt = conn.createStatement();

    if(conn!=null)
        System.out.println("Connection successful..");
    do
    {
    System.out.println("1: View Records");
    System.out.println("2: Insert Record");
    System.out.println("3: Delete Record");
    System.out.println("4: Modify Record");
    System.out.println("5: Search Record");
    System.out.println("6: Exit");
    System.out.println("\nEnter your choice : ");
    choice = Integer.parseInt(br.readLine());
    switch(choice)
    {
        case 1:
        rs = stmt.executeQuery("Select * from student");
        while(rs.next())
 {
                System.out.print("Roll Number = " + rs.getInt(1));
                System.out.println("Name = " + rs.getString(2));
            }
        break;
    case 2:
        System.out.println("Enter the roll number");
        r = Integer.parseInt(br.readLine());
        System.out.println("Enter Name:");
        name = br.readLine();
                System.out.println("Enter Percentage:");
        per = Float.parseFloat(br.readLine());
        ps1 = conn.prepareStatement("Insert into student values(?,?,?)");
        ps1.setInt(1,r);
                ps1.setString(2,name);
                ps1.setFloat(3, per);
                ps1.executeUpdate();
                System.out.println("record inserted successfully");
        break;
    case 3:
        System.out.println("Enter the roll number to be deleted ");
        r = Integer.parseInt(br.readLine());
        stmt.executeUpdate("Delete from student where rollno = " + r);
                System.out.println("record deleted successfully");
        break;
    case 4:
        System.out.println("Enter the roll number to be modified ");
        r = Integer.parseInt(br.readLine());
        System.out.println("Enter new name");
        name = br.readLine();
                System.out.println("Enter new perctentage");
        per = Float.parseFloat(br.readLine());
    ps2 = conn.prepareStatement("Update student set name = ?,percentage=? where 
rollno = ?");
                ps2.setString(1,name);
                ps2.setFloat(2,per);
                ps2.setInt(3,r);
        ps2.executeUpdate();
                 System.out.println("record modified successfully");
        break;
    case 5:
        System.out.println("Enter the roll number to be searched ");
        r = Integer.parseInt(br.readLine());
        rs = stmt.executeQuery("Select * from student where rollno = " + r);
        if(rs.next())
        {
            System.out.print("Roll Number = " + rs.getInt(1));
                System.out.println("Name = " + rs.getString(2));
                        System.out.println("Percentage = " + rs.getFloat(3));
        }
        else
            System.out.println("Student not found");
        break;
     }
   } while(choice != 6);
   }
}
```

SET B .
b) Design a following Registration form and raise an appropriate exception if invalid information is entered like Birth Year '0000'

```
// Java program to implement
// a Simple Registration Form
// using Java Swing

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

class MyFrame
    extends JFrame
    implements ActionListener {

    // Components of the Form
    private Container c;
    private JLabel title;
    private JLabel name;
    private JTextField tname;
    private JLabel mno;
    private JTextField tmno;
    private JLabel gender;
    private JRadioButton male;
    private JRadioButton female;
    private ButtonGroup gengp;
    private JLabel dob;
    private JComboBox date;
    private JComboBox month;
    private JComboBox year;
    private JLabel add;
    private JTextArea tadd;
    private JCheckBox term;
    private JButton sub;
    private JButton reset;
    private JTextArea tout;
    private JLabel res;
    private JTextArea resadd;

    private String dates[]
            = { "1", "2", "3", "4", "5",
                    "6", "7", "8", "9", "10",
                    "11", "12", "13", "14", "15",
                    "16", "17", "18", "19", "20",
                    "21", "22", "23", "24", "25",
                    "26", "27", "28", "29", "30",
                    "31" };
    private String months[]
            = { "Jan", "feb", "Mar", "Apr",
                    "May", "Jun", "July", "Aug",
                    "Sup", "Oct", "Nov", "Dec" };
    private String years[]
            = { "1995", "1996", "1997", "1998",
                    "1999", "2000", "2001", "2002",
                    "2003", "2004", "2005", "2006",
                    "2007", "2008", "2009", "2010",
                    "2011", "2012", "2013", "2014",
                    "2015", "2016", "2017", "2018",
                    "2019" };

    // constructor, to initialize the components
    // with default values.
    public MyFrame()
    {
            setTitle("Registration Form");
            setBounds(300, 90, 900, 600);
            setDefaultCloseOperation(EXIT_ON_CLOSE);
            setResizable(false);

            c = getContentPane();
            c.setLayout(null);

            title = new JLabel("Registration Form");
            title.setFont(new Font("Arial", Font.PLAIN, 30));
            title.setSize(300, 30);
            title.setLocation(300, 30);
            c.add(title);

            name = new JLabel("Name");
            name.setFont(new Font("Arial", Font.PLAIN, 20));
            name.setSize(100, 20);
            name.setLocation(100, 100);
            c.add(name);

            tname = new JTextField();
            tname.setFont(new Font("Arial", Font.PLAIN, 15));
            tname.setSize(190, 20);
            tname.setLocation(200, 100);
            c.add(tname);

            mno = new JLabel("Mobile");
            mno.setFont(new Font("Arial", Font.PLAIN, 20));
            mno.setSize(100, 20);
            mno.setLocation(100, 150);
            c.add(mno);

            tmno = new JTextField();
```

```
tmno.setFont(new Font("Arial", Font.PLAIN, 15));
tmno.setSize(150, 20);
tmno.setLocation(200, 150);
c.add(tmno);

gender = new JLabel("Gender");
gender.setFont(new Font("Arial", Font.PLAIN, 20));
gender.setSize(100, 20);
gender.setLocation(100, 200);
c.add(gender);

male = new JRadioButton("Male");
male.setFont(new Font("Arial", Font.PLAIN, 15));
male.setSelected(true);
male.setSize(75, 20);
male.setLocation(200, 200);
c.add(male);

female = new JRadioButton("Female");
female.setFont(new Font("Arial", Font.PLAIN, 15));
female.setSelected(false);
female.setSize(80, 20);
female.setLocation(275, 200);
c.add(female);

gengp = new ButtonGroup();
gengp.add(male);
gengp.add(female);

dob = new JLabel("DOB");
dob.setFont(new Font("Arial", Font.PLAIN, 20));
dob.setSize(100, 20);
dob.setLocation(100, 250);
c.add(dob);

date = new JComboBox(dates);
date.setFont(new Font("Arial", Font.PLAIN, 15));
date.setSize(50, 20);
date.setLocation(200, 250);
c.add(date);

month = new JComboBox(months);
month.setFont(new Font("Arial", Font.PLAIN, 15));
month.setSize(60, 20);
month.setLocation(250, 250);
c.add(month);

year = new JComboBox(years);
year.setFont(new Font("Arial", Font.PLAIN, 15));
year.setSize(60, 20);
year.setLocation(320, 250);
c.add(year);

add = new JLabel("Address");
add.setFont(new Font("Arial", Font.PLAIN, 20));
```

```
        add.setSize(100, 20);
        add.setLocation(100, 300);
        c.add(add);

        tadd = new JTextArea();
        tadd.setFont(new Font("Arial", Font.PLAIN, 15));
        tadd.setSize(200, 75);
        tadd.setLocation(200, 300);
        tadd.setLineWrap(true);
        c.add(tadd);

        term = new JCheckBox("Accept Terms And Conditions.");
        term.setFont(new Font("Arial", Font.PLAIN, 15));
        term.setSize(250, 20);
        term.setLocation(150, 400);
        c.add(term);

        sub = new JButton("Submit");
        sub.setFont(new Font("Arial", Font.PLAIN, 15));
        sub.setSize(100, 20);
        sub.setLocation(150, 450);
        sub.addActionListener(this);
        c.add(sub);

        reset = new JButton("Reset");
        reset.setFont(new Font("Arial", Font.PLAIN, 15));
        reset.setSize(100, 20);
        reset.setLocation(270, 450);
        reset.addActionListener(this);
        c.add(reset);

        tout = new JTextArea();
        tout.setFont(new Font("Arial", Font.PLAIN, 15));
        tout.setSize(300, 400);
        tout.setLocation(500, 100);
        tout.setLineWrap(true);
        tout.setEditable(false);
        c.add(tout);

        res = new JLabel("");
        res.setFont(new Font("Arial", Font.PLAIN, 20));
        res.setSize(500, 25);
        res.setLocation(100, 500);
        c.add(res);

        resadd = new JTextArea();
        resadd.setFont(new Font("Arial", Font.PLAIN, 15));
        resadd.setSize(200, 75);
        resadd.setLocation(580, 175);
        resadd.setLineWrap(true);
        c.add(resadd);

        setVisible(true);
    }
    // method actionPerformed()
    // to get the action performed
    // by the user and act accordingly
    public void actionPerformed(ActionEvent e)
    {
        if (e.getSource() == sub) {
            if (term.isSelected()) {
                String data1;
                String data
                    = "Name : "
                    + tname.getText() + "\n"
                    + "Mobile : "
                    + tmno.getText() + "\n";
                if (male.isSelected())
                    data1 = "Gender : Male"
                            + "\n";
                else
                    data1 = "Gender : Female"
                            + "\n";
                String data2
                    = "DOB : "
                    + (String)date.getSelectedItem()
                    + "/" + (String)month.getSelectedItem()
                    + "/" + (String)year.getSelectedItem()
                    + "\n";

                String data3 = "Address : " + tadd.getText();
                tout.setText(data + data1 + data2 + data3);
                tout.setEditable(false);
                res.setText("Registration Successfully..");
            }
            else {
                tout.setText("");
                resadd.setText("");
                res.setText("Please accept the"
                            + " terms & conditions..");
            }
        }

        else if (e.getSource() == reset) {
            String def = "";
            tname.setText(def);
            tadd.setText(def);
            tmno.setText(def);
            res.setText(def);
            tout.setText(def);
            term.setSelected(false);
            date.setSelectedIndex(0);
            month.setSelectedIndex(0);
            year.setSelectedIndex(0);
            resadd.setText(def);
        }
    }
}
```

```
// Driver Code
class Registration {

    public static void main(String[] args) throws Exception
    {
             MyFrame f = new MyFrame();
    }
}
```

SET C

a) Create tables: Course (courseid, coursename, course instructor) and Student (studentid, studentname, studentclass). Course and Student have a many to many relationship. Create a GUI based system for performing the following operations on

the tables: Course: Add Course, View All students of a specific course Student: Add Student, Delete Student, View All students, Search student.

```
// Java program to implement a GUI
// application for the student
// management system

import javax.swing.*;
import java.awt.*;
import java.awt.Image;
import java.awt.event.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.print.*;
import javafx.print.Printer;
import java.io.*;
import java.io.IOException;

// Creating the fee class
public class fee extends Frame {

    JLabel l1, l2, l3, l4,
             l5, l6, l7, l8,
             l9, l10, l12, l13,
             l14, l11, l15;

    JTextField tf1, tf2, tf3,
             tf4, tf5, tf6,
             tf7, tf8, tf9,
             tf10;

    JTextArea area2, area1;

    JRadioButton rb1, rb2, rb3,
             rb4, rb5, rb6,
             rb7;

    JFileChooser f1;

    // Default constructor to
    // initialize the parameters
    fee()
    {

             l1 = new JLabel("Fee Report");
             l1.setBounds(550, 100, 250, 20);

             l2 = new JLabel(
                     "Name of the Student:");
             l2.setBounds(50, 150, 250, 20);

             tf1 = new JTextField();
             tf1.setBounds(250, 150, 250, 20);

             l3 = new JLabel(
                     "Name of the Father:");
             l3.setBounds(50, 200, 250, 20);

             tf2 = new JTextField();
             tf2.setBounds(250, 200, 250, 20);

             l4 = new JLabel("Roll Number:");
             l4.setBounds(50, 250, 250, 20);

             tf3 = new JTextField();
             tf3.setBounds(250, 250, 250, 20);

             l5 = new JLabel("Email ID:");
             l5.setBounds(50, 300, 250, 20);

             tf4 = new JTextField();
             tf4.setBounds(250, 300, 250, 20);

             l6 = new JLabel("Contact Number:");
             l6.setBounds(50, 350, 250, 20);

             tf5 = new JTextField();
             tf5.setBounds(250, 350, 250, 20);

             l7 = new JLabel("Address:");
             l7.setBounds(50, 400, 250, 20);

             area1 = new JTextArea();
             area1.setBounds(250, 400, 250, 90);

             l9 = new JLabel("Gender:");
             l9.setBounds(50, 500, 250, 20);

             JRadioButton r5
                     = new JRadioButton(" Male");
             JRadioButton r6
                     = new JRadioButton(" Female");

             r5.setBounds(250, 500, 100, 30);
             r6.setBounds(350, 500, 100, 30);
```

```
ButtonGroup bg = new ButtonGroup();
bg.add(r5);
bg.add(r6);

l10 = new JLabel("Nationality:");
l10.setBounds(50, 550, 250, 20);

tf6 = new JTextField();
tf6.setBounds(250, 550, 250, 20);

l11 = new JLabel(
        "Year of passing 10th");
l11.setBounds(50, 600, 250, 20);

String language[]
        = { "2016", "2015", "2014" };

final JComboBox cb1
        = new JComboBox(language);

cb1.setBounds(250, 600, 90, 20);

l12 = new JLabel(
        "Year of passing 12th");
l12.setBounds(50, 650, 250, 20);

String languagess[]
        = { "2019", "2018", "2017" };

l13 = new JLabel(
        "Points Secured in 10th:");
l13.setBounds(50, 700, 250, 20);

tf7 = new JTextField();
tf7.setBounds(250, 700, 250, 20);

l14 = new JLabel("Percentage in 12th:");
l14.setBounds(50, 750, 250, 20);

tf8 = new JTextField();
tf8.setBounds(250, 750, 250, 20);

ImageIcon i2 = new ImageIcon("2.png");
JLabel l15
        = new JLabel("", i2, JLabel.CENTER);

l15.setBounds(900, 50, 600, 200);

final JComboBox cb2
        = new JComboBox(languagess);

cb2.setBounds(250, 650, 90, 20);
l8 = new JLabel(
        "Groups Offered here are:");
```

```
l8.setBounds(800, 150, 250, 20);

rb1 = new JRadioButton("SEAS");
rb1.setBounds(550, 150, 100, 30);

rb2 = new JRadioButton("SLABS");
rb2.setBounds(650, 150, 100, 30);

ButtonGroup bg1 = new ButtonGroup();

bg1.add(rb1);
bg1.add(rb2);

rb3 = new JRadioButton("HOSTELLER");
rb3.setBounds(550, 200, 100, 30);

rb4 = new JRadioButton("DAY SCHOLAR");
rb4.setBounds(650, 200, 120, 30);

ButtonGroup bg2 = new ButtonGroup();
bg2.add(rb3);
bg2.add(rb4);

String languages[]
        = { "CSE", "ECE", "EEE",
                "CIVIL", "MECH" };
final JComboBox cb
        = new JComboBox(languages);
cb.setBounds(800, 200, 90, 20);

final JLabel label
        = new JLabel();
label.setBounds(600, 430, 500, 30);
JButton b = new JButton("Show");
b.setBounds(1000, 300, 80, 30);

final DefaultListModel<String> li1
        = new DefaultListModel<>();

li1.addElement("CSE(2, 50, 000)");
li1.addElement("ECE(2, 50, 000)");
li1.addElement("EEE(2, 50, 000)");
li1.addElement("MECH(2, 50, 000)");
li1.addElement("CIVIL(2, 50, 000)");

final JList<String> list1
        = new JList<>(li1);

list1.setBounds(600, 300, 125, 125);

DefaultListModel<String> li2
        = new DefaultListModel<>();

li2.addElement(
        "2 SHARE(1, 50, 000)");
```

```
li2.addElement(
        "3 SHARE(1, 40, 000)");
li2.addElement(
        "5 SHARE(1, 20, 000)");
li2.addElement(
        "8 SHARE(1, 10, 000)");
li2.addElement(
        "bus(40, 000)");

final JList<String> list2
        = new JList<>(li2);
list2.setBounds(
        800, 300, 125, 125);

JButton Receipt
        = new JButton("Generate Receipt");
Receipt.setBounds(600, 490, 150, 30);
JButton b2 = new JButton("Reset");
b2.setBounds(750, 490, 150, 30);
JButton Print = new JButton("Print");
Print.setBounds(900, 490, 150, 30);

area2 = new JTextArea();
area2.setBounds(600, 540, 450, 240);

add(l1);
add(l2);
add(l3);
add(l4);
add(l5);
add(l6);
add(l7);
add(l8);
add(l9);
add(l10);
add(l11);
add(l12);
add(l13);
add(l14);
add(tf1);
add(tf2);
add(tf3);
add(tf4);
add(tf5);
add(tf6);
add(tf7);
add(tf8);
add(area1);
add(area2);
add(l15);
add(rb1);
add(rb2);
add(rb3);
add(rb4);
add(r5);
add(r6);
add(cb);
add(cb1);
add(cb2);
add(list1);
add(list2);
add(b);
add(label);
add(Receipt);
add(b2);
add(Print);

b.addActionListener(new ActionListener() {

        // Method to display the data
        // entered in the text fields
        public void actionPerformed(ActionEvent e)
        {
                String data = "";
                if (list1.getSelectedIndex() != -1) {
                        data = "You had selected the Group:"
                                + list1.getSelectedValue();
                        label.setText(data);
                }
                if (list2.getSelectedIndex() != -1) {
                        data += " and Hostel with the "
                                        + "facility of: ";

                        for (Object frame :
                                list2.getSelectedValues()) {
                                data += frame + " ";
                        }
                }
                label.setText(data);
        }
});

// Reset the text fields
b2.addActionListener(
        new ActionListener() {
                public void actionPerformed(
                        ActionEvent e)
                {
                        area2.setText("");
                        area1.setText(" ");
                        tf1.setText("");
                        tf2.setText("");
                        tf3.setText("");
                        tf4.setText("");
                        tf5.setText("");
                        tf6.setText(" ");
                }
        });

// Implementing the Print action
```

```
Print.addActionListener(
        new ActionListener() {
                public void actionPerformed(
                        ActionEvent e)
                {
                        try {
                                area2.print();
                        }
                        catch (java.awt.print
                                        .PrinterException a) {
                                System.err.format(
                                        "NoPrinter Found",
                                        a.getMessage());
                        }
                }
        });

// Generating the receipt
Receipt.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e)
        {

                area2.setText(
                        "--------------------------------"
                        + "-----------FEE RECEIPT----"
                        + "--------------------------"
                        + "--------------------------"
                        + "-------------------\n");

                area2.setText(area2.getText()
                                        + "Student Name: "
                                        + tf1.getText()
                                        + "\n");
                area2.setText(area2.getText()
                                        + "Father's Name: "
                                        + tf2.getText()
                                        + "\n");
                area2.setText(area2.getText()
                                        + "RollNumber: "
                                        + tf3.getText()
                                        + "\n");
                area2.setText(area2.getText()
                                        + "Email ID: "
                                        + tf4.getText()
                                        + "\n");
                area2.setText(area2.getText()
                                        + "Contact Number: "
                                        + tf5.getText()
                                        + "\n");
                area2.setText(area2.getText()
                                        + "Wants to take: "
                                        + cb.getSelectedItem()
                                                        .toString()
                                        + "\n");
```

```
if (rb1.isSelected()) {
        area2.setText(area2.getText()
                        + "Wants to Join in "
                        + "School of Engineering "
                        + "and Applied Sciences\n");
}
if (rb2.isSelected()) {
        area2.setText(area2.getText()
                        + "Wants to Join in "
                        + "School of Liberal "
                        + "Arts and Sciences\n");
}
if (rb3.isSelected()) {
        area2.setText(area2.getText()
                        + "Wants to be a "
                        + "Hosteller \n");
}
if (rb4.isSelected()) {
        area2.setText(area2.getText()
                        + "Wants to be a "
                        + "Day Scholar \n");
}
area2.setText(area2.getText()
                + "Had chosen: "
                + list1.getSelectedValue()
                        .toString()
                + "\n");
area2.setText(area2.getText()
                + "Had chosen: "
                + list2.getSelectedValue()
                        .toString()
                + "\n");

int index2 = list2.getSelectedIndex();
if (index2 == 0) {
        area2.setText(area2.getText()
                        + "                              "
                        + "Total amount to be "
                        + "paid is 4 Lakhs \n");
}

if (index2 == 1) {
        area2.setText(area2.getText()
                        + "                              "
                        + "Total amount to be paid "
                        + "is 3.9 Lakhs \n");
}

if (index2 == 2) {
        area2.setText(area2.getText()
                        + "                              "
                        + "Total amount to be paid "
                        + "is 3.8 Lakhs \n");
}
```

```
                        if (index2 == 3) {
                            area2.setText(area2.getText()
                                            + "                    "
                                            + "Total amount to be paid "
                                            + "is 3.7 Lakhs \n");
                        }

                        if (index2 == 4) {
                            area2.setText(area2.getText()
                                            + "                    "
                                            + "Total amount to be paid "
                                            + "is 2.9 Lakhs \n");
                        }

                        if (e.getSource() == Receipt) {
                            try {
                                FileWriter fw
                                    = new FileWriter(
                                        "java.txt", true);
                                fw.write(area2.getText());
                                fw.close();
                            }
                            catch (Exception ae) {
                                System.out.println(ae);
                            }
                        }

                        JOptionPane.showMessageDialog(
                            area2, "DATA SAVED SUCCESSFULLY");
                    };
                });
            addWindowListener(
                new WindowAdapter() {
                    public void windowClosing(
                        WindowEvent we)
                    {
                        System.exit(0);
                    }
                });
            setSize(800, 800);
            setLayout(null);
            setVisible(true);
            setBackground(Color.cyan);
    }
    public static void main(String[] args)
    {
            new fee();
    }
}
```

Ass 4
SET A
a) Design a servlet that provides information about a HTTP request from a client, such as IP address and browser type. The servlet also provides information about the server on which the servlet is running, such as the operating system type, and the names of currently loaded servlets.

serverInfo.java

```
    import java.io.*;
    import javax.servlet.*;
    import javax.servlet.http.*;
public class serverInfo extends HttpServlet implements Servlet
{
  protected void doGet(HttpServletRequest req,HttpServletResponse res)throws
IOException,ServletException
  {
    res.setContentType("text/html");
    PrintWriter pw=res.getWriter();
    pw.println("<html><body><h2>Information about Http Request</h2>");
    pw.println("<br>Server Name: "+req.getServerName());
    pw.println("<br>Server Port: "+req.getServerPort());
    pw.println("<br>Ip Address: "+req.getRemoteAddr());
//pw.println("<br>Server Path: "+req.getServerPath());    pw.println("<br>Client Browser:
"+req.getHeader("User-Agent"));
    pw.println("</body></html>");
    pw.close();
  }
}
```

Web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app>
<servlet>
<servlet-name>serverInfo</servlet-name>
<servlet-class>ServerInfo</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>serverInfo</servlet-name>
<url-pattern>/server</url-pattern>
</servlet-mapping>
</web-app>
```