ELECTRICITY BILL GENERATION

```
import java.io.*;
import java.util.*;
class EbBill
   {
      int custNo;
      String custName;
      int prevMonthReading;
      int currMonthReading;
      String connType;
      int units;
      double billAmount;
       public void getData()
      {
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter the Customer Number          :");
        custNo = sc.nextInt();
        System.out.print("Enter the Customer Name            :");
        custName = sc.next();
        System.out.print("Enter the  Previous Month Reading   :");
        prevMonthReading = sc.nextInt();
        System.out.print("Enter the  Current Month Reading    :");
        currMonthReading = sc.nextInt();
        System.out.print("Enter the EB Connection Type        :");
        connType = sc.next();
    }
    public void computeBill()
    {
    if ((connType.equals("d")) ||(connType.equals("D")))
      {
        units = currMonthReading - prevMonthReading;
         if(units < 100)
```

```
            billAmount = units * 1.00;
        else if(units <= 200)
        billAmount = 100*1.00+(units-100)*2.50;
        else if(units <= 500)
        billAmount = 100*1.00+100*2.50+(units-200)*4.00;
        else if(units > 500)
          billAmount = 100*1.00+100*2.50+300*4+(units-500)*6.00;
      }
    else if ((connType.equals("c")) ||(connType.equals("C")) )
      {
       units = currMonthReading - prevMonthReading;
        if(units < 100)
        billAmount = units * 2.00;
        else if(units <= 200)
        billAmount = 100*2.00+(units-100)*4.50;
        else if(units <= 500)
        billAmount = 100 *2.00+100*4.50+(units-200)*6.00;
        else if(units > 500)
        billAmount = 100*2.00+100*4.50+300*6+(units-500)*7.00;
    }    }
  public void displayBill()
    {
    System.out.println("Electricity Bill");
    System.out.println("Customer Number            : " + custNo);
    System.out.println("Customer Name              : " + custName);
    System.out.println("Previous Month Reading     : " +
prevMonthReading);
    System.out.println("Current Month Reading      : " +
currMonthReading);
    System.out.println("EB Connection Type         : " + connType);
    System.out.println("Number of Units Consumed   : " + units);
    System.out.println("Amount to be Paid          : " + billAmount);
    }   }
```

```
    class EbBillGeneration
{
public static void main(String []args)
{
    EbBill eb = new EbBill();
    eb.getData();
    eb.computeBill();
    eb.displayBill();
}
}
```

CURRENCY CONVERTER

// Currency.java

```
package Currconvert;
import java.io.*;
import java.util.*;
public class Currency
   {
        double inRupee,inDollar,inEuro,inYen;
        double conversion;
        double oneDollarRs = 68.59;
        double oneEuroRs = 79.78;
        double oneYenRs = 0.62;
        double oneRsEuro = 0.013;
        double oneRsDollar = 0.015;
        double oneRsYen = 1.61;
        public void dollerConversion()
       {
         Scanner sc=new Scanner(System.in);
         System.out.print("Enter the amount in Indian Rupees(₹):");
         inRupee = sc.nextDouble();
         System.out.println("Amount in United States Dollar($):" +
          String.format("%.2f",inRupee*oneRsDollar));
         System.out.print("Enter the amount in United States Dollar($):");
         inDollar = sc.nextDouble();
         System.out.println("Amount in Indian Rupees(₹):" +
          String.format("%.2f",inDollar*oneDollarRs));
       }
        public void euroConversion()
       {
         Scanner sc=new Scanner(System.in);
         System.out.print("Enter the amount in Indian Rupees(₹):");
         inRupee = sc.nextDouble();
```

```
        System.out.println("Amount in Euro(€):" +
       String.format("%.2f",inRupee*oneRsEuro));
        System.out.print("Enter the amount in Euro(€):");
       inEuro = sc.nextDouble();
        System.out.println("Amount in Indian Rupees(₹)" +
       String.format("%.2f",inEuro*oneEuroRs));
    }
     public void yenConversion()
    {
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter the amount in Indian Rupees(₹):");
        inRupee = sc.nextDouble();
      System.out.println("Amount in Yen(¥):" +
        String.format("%.2f",inRupee*oneRsYen));
      System.out.print("Enter the amount in Yen(¥):");
        inYen = sc.nextDouble();
      System.out.println("Amount in Indian Rupees(₹):" +
        String.format("%.2f",inYen*oneYenRs));
    }
    }
```

// Converter.java

```
import Currconvert.Currency;
class Converter
{
  public static void main(String []args)
     {
        Currency c = new Currency();
        c.dollerConversion();
        c.euroConversion();
        c.yenConversion();
    }}
```

DISTANCE CONVERTER

// DistanceConverter.java

```
package distanceconverter;
import java.util.Scanner;
public class DistanceConverter
{
    static double convertIntoKms(double miles)
    {
    double km=1.609*miles;
     return km;
    }
     static double convertIntoMiles(double km)
    {
     double miles=km/1.609;
        return miles;
    }
   }
```

//Converter.java

```
import distanceconverter.DistanceConverter
public class Converter
{
public static void main(String[] args)
  {
DistanceConverter d = new DistanceConverter();
Scanner sc  =   new Scanner(System.in);
System.out.print("Enter Distance in Miles  : ");
double miles   =   sc.nextDouble();
 System.out.println(miles+" Miles equal to : "+d.convertIntoKms(miles)+"
KMs");
 System.out.print("Enter Distance in Km  : ");
  double kms   =   sc.nextDouble();
```

```
  System.out.println(kms+" KMs equal to : "+d.convertIntoMiles(kms)+"
Miles")
     }   }
```

TIME CONVERTER

// TimeConverter.java

```
package timeconverter;
import java.util.Scanner;
public class TimeConverter
{
   double convertIntoMin(double hr)
   {
     return 60*hr;
    }
    double convertIntoSec(double hr)
    {
      return 3600*hr;
      }
   double convertMinIntoHr(double min)
     {
     return min/60;
      }
    double convertSecIntoHr(double hr)
     {
        return hr/3600;
        }
  }
```

// Converter.java

```
import timeconverter.TimeConverter;
class Converter
{
```

```
 public static void main(String []args)
{
        TimeConverter t= new TimeConverter();
        Scanner sc  =   new Scanner(System.in);
        System.out.print("Enter Time in Hours  : ");
        double hrs   =   sc.nextInt();
        System.out.println(hrs+" Hours equal to : "+convertIntoMin(hrs)+"
Minutes");
        System.out.println(hrs+" Hours equal to : "+convertIntoSec(hrs)+"
Seconds");
        System.out.print("Enter Time in Minutes  : ");
        double min   =   sc.nextInt();
        System.out.println(min+" Minutes equal to :
"+convertMinIntoHr(min)+" Hours");
        System.out.print("Enter Time in Seconds  : ");
        double sec   =   sc.nextInt();
        System.out.println(sec+" Seconds equal to : "+convertSecIntoHr(sec)+"
Hours");
}
}
```

PAY SLIP GENERATION USING INHERITANCE

```
import java.io.*;
import java.util.*;
class Employee
   {
        int empId;
        String empName;
        String address;
        String emailId;
        long mobileNo;
        float da;
        float hra;
        float pf;
        float staffFund;
        float grossSalary;
        float netSalary;
        public void readDetails()
       {
          Scanner sc=new Scanner(System.in);
          System.out.print("Enter the Employee ID    :");
          empId = sc.nextInt();
          System.out.print("Enter the Employee Name   :");
          empName = sc.next();
          System.out.print("Enter the  Address       :");
          address = sc.next();
          System.out.print("Enter the  E-Mail id   :");
          emailId = sc.next();
          System.out.print("Enter the Mobile Number    :");
          mobileNo = sc.nextInt();
       }

       public void computeSalary(float bp)
       {
```

```
            da = (bp * 97) / 100;
            hra = (bp * 10) / 100;
            pf = (bp * 12) / 100;
            staffFund = (bp * 1/10) / 100;
            grossSalary = bp + da + hra;
            netSalary = grossSalary - (pf+staffFund);
        }
      public void displayPaySlip(float bp, String designation)
    {
        System.out.println("Pay Slip");
        System.out.println("Employee ID          : " + empId);
        System.out.println("Employee Name      : " + empName);
        System.out.println("Address                 : " + address);
        System.out.println("E-Mail ID               : " + emailId);
        System.out.println("Mobile Number       : " + mobileNo);
         System.out.println("Designation          : " + designation);
        System.out.println("Basic Pay                 : " + bp);
        System.out.println("Dearness Allowance(DA)      : " + da);
        System.out.println("Housse Rent Allowance(HRA)   : " + hra);
        System.out.println("Provident Fund(PF)      : " + pf);
        System.out.println("Staff Club Fund          : " + staffFund);
         System.out.println("Gross Pay               : " + grossSalary);
         System.out.println("Net Pay                 : " + netSalary);
    }
}
class Programmer extends Employee
    {
            float basicPay = 5000;
            String designation = "Programmer";
    }
class AsstProf extends Employee
    {
            float basicPay = 10000;
```

```
                String designation = "Assistant Professor";
        }
class AssoProf extends Employee
        {
                float basicPay = 15000;
                String designation = "Associate Professor";
        }
class Prof extends Employee
        {
                float basicPay = 20000;
                String designation = "Professor";
        }
  class PaySlipGeneration
{
    public static void main(String []args)
    {
        int choice;
        Scanner sc = new Scanner(System.in);
        do
        {
        System.out.print("1. Programmer 2. Assistant Professor 3. Associate
Professor
                        4. Professor\n Enter your Choice :");
        choice = sc.nextInt();
        switch(choice)
                {
                case 1 :
                        {
                        Programmer prog = new Programmer();
                        prog.readDetails();
                        prog.computeSalary(prog.basicPay);
                        prog.displayPaySlip(prog.basicPay, prog.designation);
                        break;
```

```
                }
                case 2 :
                        {
                        AsstProf ap = new AsstProf();
                        ap.readDetails();
                        ap.computeSalary(ap.basicPay);
                        ap.displayPaySlip(ap.basicPay, ap.designation);
                        break;
                        }
                case 3 :
                        {
                        AssoProf asp = new AssoProf();
                        asp.readDetails();
                        asp.computeSalary(asp.basicPay);
                        asp.displayPaySlip(asp.basicPay,asp.designation);
                        break;
                        }
                case 4 :
                        {
                        Prof p = new Prof();
                        p.readDetails();
                        p.computeSalary(p.basicPay);
                        p.displayPaySlip(p.basicPay, p.designation);
                        break;
                        }
                default:
                        {
                                System.out.println("Invalid Choice");
                        }
                }
    }while(choice<=4);
  }
}
```

STACK ADT USING INTERFACE AND EXCEPTION

```
import java.io.*;
import java.util.*;
interface Stack1
{
void push(int item);//pushes the element
int pop();//pops the element
}
class StackInt implements Stack1
{
private int stck[];
private int tos;
StackInt(int size)
{
stck=new int[size];
tos=-1;
}
public void push(int item)
{
if(tos==stck.length-1)
System.out.println("Stack is full");
else
stck[++tos]=item;
}
public int pop()
{
if(tos<0)
{
System.out.println("Stack underflow");
return 0;
}
else
```

```
return stck[tos--];
}}
class Tst
{
public static void main(String args[])
{
StackInt mystack=new StackInt(10);
try
{
for(int i=0;i<5;i++)
mystack.push(i);
System.out.println("The elemenets in Stack are:");
for(int i=0;i<15;i++)
System.out.println(mystack.pop());}
catch(ArrayIndexOutOfBoundsException e)
{
System.out.println("Stack out of bounds error"+e);
}}
}
```

ARRAY LIST

```
class Operations
{
void append(ArrayList a,String s)
{
a.add(s);
}
void append(ArrayList a, int pos, String s)
{
a.add(pos,s);
}
void removeElement(ArrayList a,int pos)
{
a.remove(pos);
}
void search(ArrayList a,String s)
{
System.out.println(a.contains(s));
}
void listElement(ArrayList a,String s)
{
for(int i=0;i<a.size();i++)
{
String s1=a.get(i).toString();
if (s1.startsWith(s))
System.out.println(a.get(i));
}}
}
class ArrayListDemo
{
public static void main (String[] args) throws java.lang.Exception
{
```

```
 ArrayList<String> al=new ArrayList<String>();
Operations op=new Operations();
op.append(al,"Asha");
System.out.println(al);
op.append(al,"Naveen");
System.out.println(al);
op.append(al,1,"Lakshmi");
System.out.println(al);
op.append(al,"Seetha");
System.out.println(al);
op.removeElement(al,2);
System.out.println(al);
op.search(al,"Seetha");
op.append(al,"Jaya");
System.out.println(al);
op.listElement(al,"J");
}
}
```

AREA CALCULATION USING ABSTRACT CLASS

```
import java.io.*;
abstract class Shape
{
private double height;  // To hold height.
private double width;   //To hold width or base
private double radius;  //To hold radius
public void setValues(double height, double width)
 {
 this.height = height;
 this.width = width;
 }
public void setValues(double radius)
 {
   this.radius = radius;
  }
public double getHeight()
 {
    return height;
 }
 public double getWidth()
{
  return width;
  }
public double getRadius()
 {
   return radius;
  }
  // The getArea method is abstract. It must be overridden in a subclass.
public abstract double area();
}
class Rectangle extends Shape
```

```
{
    //Calculate and return area of rectangle
    public double area()
    {
     return getHeight() * getWidth();
    }}
// This class Triangle calculates the area of triangle
class Triangle extends Shape
{
   //Calculate and return area of triangle
    public double area()
    {
     return (getHeight() * getWidth()) / 2;
    } }
// This class Circle calculates the area of circle
class Circle extends Shape
{
    //Calculate and return area of rectangle
    public double area()
    {
         double r = getRadius();
          return ((22.0/7.0) * r * r);
    }}
// This classs demonstrates polymorphic behavior.
public class AbstractDemo
{
    public static void main(String[] args)
    {
        Shape shape;
        // assign subclass reference to subclass variable
        Rectangle rect = new Rectangle();
         // shape points to the object rect.
        shape = rect;
```

```
 // Set data in Rectangle's object
        shape.setValues(78, 5);
       //Display the area of rectangle
        System.out.println("Area of Rectangle : " + shape.area());
         // assign subclass reference to subclass variable
        Triangle tri = new Triangle();
      // shape points to the object rect.
        shape = tri;
    // Set data in Triangle's object
        shape.setValues(34,3);

//Display the area of triangle
         System.out.println("Area of Triangle : " + shape.area());
         Circle cir = new Circle();

// shape points to the object cir.
       shape = cir;

// Set data in circle's object
        shape.setValues(5);

//Display the area of circle
        System.out.println("Area of Circle : " + shape.area());
}}
```

CUSTOM EXCEPTION HANDLING

```
import java.io.*;
import java.util.*;
class StackException extends Exception
{
public String toString(){
return "Stack Underflow/Overflow";
}
}
interface Stack1
{
void push(int item) throws StackException;//pushes the element
int pop() throws StackException;//pops the element
}
class StackInt implements Stack1
{
private int stck[];
private int tos;
StackInt(int size)
{
stck=new int[size];
tos=-1;
}
public void push(int item) throws StackException
{
if(tos==stck.length-1)
throw new StackException();
stck[++tos]=item;
}
public int pop() throws StackException
{
if(tos<0)
{
```

```
throw new StackException();
}
return stck[tos--];
}
}
class stackEx
{
public static void main(String args[])
{
try{
StackInt mystack=new StackInt(10);
for(int i=0;i<5;i++)
mystack.push(i);
System.out.println("The elemenets in Stack are:");
for(int i=0;i<15;i++)
System.out.println(mystack.pop());
}
catch(StackException e)
{
System.out.println("Error detected: " +e.toString() );
}
}
}
```

FILE

```
import java.io.*;
import java.lang.*;
import java.util.*;
import java.io.File;
public class FileDemo
{
   public static void main(String[] args)
{
      File f = null;
      boolean bool = false;
      try
    {
    Scanner sc=new Scanner(System.in);
    System.out.print("Enter a File Name          : ");
    String fileName  = sc.next();

 // create new files
     f = new File("//home/exam1/exam1001/" + fileName);

 // tests if file exists
   bool = f.exists();

 // prints
     System.out.println("File exists: " + bool);

  // create new file in the system
     f.createNewFile();

 // tests if file exists
      bool = f.exists();
      if(bool == true)
    {
```

```
// printing the permissions associated with the file
 System.out.println(fileName + " is Readable: " + f.canRead());
System.out.println(fileName + " is Writable: "+ f.canWrite());
}
  else
{
   System.out.println("File not found.");
}
   // prints
System.out.println("File exists: "+bool);
long len = f.length();
String path = f.getPath();
System.out.println("File Path is " + path + " File length: " +len
}
catch(Exception e)
{
// if any error occurs
     e.printStackTrace();
    }  }}
```

MULTITHREADING

```
import java.io.*;
import java.lang.*;
import java.util.*;
class even implements Runnable
{
public int x;
public even(int x)
{
this.x = x;
}
public void run()
{
System.out.println("New Thread "+ x +" is EVEN and Square of " + x + " is: "
+ x * x);
}}
class odd implements Runnable
{
public int x;
public odd(int x)
{
this.x = x;
}
public void run()
{
System.out.println("New Thread "+ x +" is ODD and Cube of " + x + " is: " + x
* x * x);
} }
class MultiThread extends Thread
{
public void run()
{
int num = 0;
```

```
Random r = new Random();
try {
for (int i = 0; i < 10; i++)
{
num = r.nextInt(100);
System.out.println("Main Thread and Generated Number is " + num);
if (num % 2 == 0)
{
Thread t1 = new Thread(new even(num));
t1.start();
}
else
{
Thread t2 = new Thread(new odd(num));
t2.start();
}
Thread.sleep(1000);
System.out.println("--------------------------------------");
}}
catch (Exception ex)
{
System.out.println(ex.getMessage());
}}}
public class MultiThreadDemo
{
public static void main(String[] args)
{
MultiThread mt = new MultiThread();
mt.start();
}}
```

GENERIC CLASS AND METHODS

```
import java.io.*;
import java.lang.*;
import java.util.*;
public class GenericMax
{
 // determines the largest of three Comparable objects
  public static <T extends Comparable<T>>
 T maximum(T x, T y, T z)
{
 T max = x;
// assume x is initially the largest
    if (y.compareTo(max) > 0)
    max = y;
// y is the largest so far
if (z.compareTo(max) > 0)
     max = z;
// z is the largest
return max;
// returns the largest object
  }
// end method maximum
public static void main(String args[])
{
System.out.printf("Maximum of %d, %d and %d is %d\n\n", 3, 4, 5,
maximum(3, 4, 5));
System.out.printf("Maximum of %.1f, %.1f and %.1f is %.1f\n\n", 6.6, 8.8,
7.7,     maximum(6.6, 8.8, 7.7));
System.out.printf("Maximum of %s, %s and %s is %s\n", "pear", "apple",
"orange", maximum("pear", "apple", "orange"));
  }
}
```

EVENT DRIVEN PROGRAMMING

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class calculatorpanel extends JPanel
{
JButton display;
JPanel panel;
double result;
String lastcommand,a;
boolean start;
public calculatorpanel()
{
setLayout(new BorderLayout());
result=0; lastcommand="=";
start=true;
display=new JButton("0");
display.setEnabled(false);
add(display,BorderLayout.NORTH);
ActionListener insert=new insertAction();
ActionListener command=new commandAction();
panel=new JPanel();
panel.setLayout(new GridLayout(4,5)); addbutton("sin",command);
addbutton("7",insert);
addbutton("8",insert);
addbutton("9",insert);
addbutton("/",command);
addbutton("cos",command);
addbutton("4",insert);
addbutton("5",insert);
addbutton("6",insert);
addbutton("*",command);
addbutton("tan",command);
```

```
addbutton("1",insert);
addbutton("2",insert);
addbutton("3",insert);
addbutton("+",command);
addbutton("sqrt",command);
addbutton("0",insert);
addbutton(".",insert);
addbutton("=",command);
addbutton("-",command);
add(panel,BorderLayout.CENTER);
}
void addbutton(String label,ActionListener listener)
{
 JButton button=new JButton(label);
button.addActionListener(listener);
panel.add(button);
}
public void calculate(double x)
{
 if(lastcommand.equals("+"))
result+=x;
else
if(lastcommand.equals("-"))
result-=x;
else
if(lastcommand.equals("*"))
result*=x;
else
if(lastcommand.equals("/"))
result/=x;
else
if(lastcommand.equals("="))
result=x;
```

```
else
if(lastcommand.equals("sin"))
result=Math.sin(Math.toRadians(x));
else
if(lastcommand.equals("cos"))
result=Math.cos(Math.toRadians(x));
else
if(lastcommand.equals("tan"))
result=Math.tan(Math.toRadians());
else
if(lastcommand.equals("sqrt"))    result=Math.sqrt(x);
display.setText(" "+result);
}
class insertAction implements ActionListener
{
public void actionPerformed(ActionEvent e)
{
String input=e.getActionCommand();
if(start)
{
display.setText(" "); start=false;
}
display.setText(display.getText()+input);
}}
class commandAction implements ActionListener
{
public void actionPerformed(ActionEvent e)
{
String command=e.getActionCommand();
if(start)
{
if(command.equals("-"))
{
```

```
display.setText(command);
start=false;
}
else
lastcommand=command;
}
else
{
calculate(Double.parseDouble(display.getText())); lastcommand=command;
start=true;
}}}}
class calculatorframe extends JFrame
{
public calculatorframe()
{
setSize(350,250);
setTitle("Calculator");
setLocationByPlatform(true);
Toolkit kit=Toolkit.getDefaultToolkit();
Image im=kit.getImage("U:\\calicon.jpg");
setIconImage(im);
calculatorpanel panel=new calculatorpanel();
add(panel);
}}
class calci
{
public static void main(String args[])
{
calculatorframe cf=new calculatorframe();
cf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
cf.setVisible(true);
}}
```