OOPSLAB

EX NO : 1 PROGRAM TO GENERATE ELECTRICITY BILL

AIM

To develop a Java application to generate Electricity bill.

PROCEDURE

1.Create a class with the following members

Consumer no., consumer name, previous month reading, current month reading, type of EB
connection (i.e domestic or commercial)

2.Compute the bill amount using the following tariff.

If the type of the EB connection is domestic, calculate the amount to be paid as follows:

- First 100 units - Rs. 1 per unit
- 101-200 units - Rs. 2.50 per unit
- 201 -500 units - Rs. 4 per unit
- > 501 units - Rs. 6 per unit

If the type of the EB connection is commercial, calculate the amount to be paid as follows:

- First 100 units - Rs. 2 per unit
- 101-200 units - Rs. 4.50 per unit
- 201 -500 units - Rs. 6 per unit
- > 501 units - Rs. 7 per unit

3. Create the object for the created class .Invoke the methods to get the input from the consumer

and display the consumer information with the generated electricity bill.

PROGRAM

```
import java.util.*;
class ebill
{
public static void main (String args[])
{
customerdata ob = new customerdata();
ob.getdata();
ob.calc();
ob.display();
}
}
class customerdata
{
Scanner in = new Scanner(System.in);
Scanner ins = new Scanner(System.in);
String cname,type;
int bn;
double current,previous,tbill,units;
void getdata()
{
System.out.print ("\n\t Enter consumer number  ");
bn = in.nextInt();
System.out.print ("\n\t Enter Type of connection  (D for Domestic or C for Commercial)
");
type = ins.nextLine();
System.out.print ("\n\t Enter consumer name  ");
cname = ins.nextLine();
System.out.print ("\n\t Enter previous month reading  ");
previous= in.nextDouble();
System.out.print ("\n\t Enter current month reading  ");
current= in.nextDouble();
}
void calc()
{
```

```
units=current-previous;
if(type.equals("D"))
{
if (units<=100)
tbill=1 * units;
else if (units>100 && units<=200)
tbill=2.50*units;
else if(units>200 && units<=500)
tbill= 4*units;
else
tbill= 6*units;
}
else
{
if (units<=100)
tbill= 2 * units;
else if(units>100 && units<=200)
tbill=4.50*units;
else if(units>200 && units<=500)
tbill= 6*units;
else
tbill= 7*units;
}
}
void display()
{
System.out.println("\n\t Consumer number = "+bn);
System.out.println ("\n\t Consumer name = "+cname);
if(type.equals("D"))
System.out.println ("\n\t type of connection  = DOMESTIC ");
else
System.out.println ("\n\t type of connection  = COMMERCIAL ");
System.out.println ("\n\t Current Month  Reading = "+current);
System.out.println ("\n\t Previous Month  Reading = "+previous);
System.out.println ("\n\t Total units = "+units);
System.out.println ("\n\t Total bill = RS "+tbill);
}
}
```

OUTPUT

RESULT

Thus the java program to generate electricity bill was implemented and executed successfully

EX NO: 2 PROGRAM TO IMPLEMENT CURRENCY CONVERTER, DISTANCE CONVERTER AND TIME CONVERTER USING PACKAGES

AIM

To develop a java application to implement currency converter ,distance converter and time converter using the concept of packages .

PROCEDURE

1. Create a Package currencyconversion and place the class currency under the package
2. Create the methods to perform currency conversion from dollar to rupee ,rupee to

dollar,euro to rupee , rupee to euro , yen to rupee and rupee to yen.

3. Create the package distanceconverion and create the class distance within the package

4. Create the methods to convert from meter to km, km to meter, miles to km,km to miles

5. Create the package timeconversion and create the class timer .Create the methods to convert from hours to minutes ,hours to seconds , minutes to hours and seconds to hours

6. Create a class and import the packages currencyconversion,distanceconversion and time conversion.Create the objects for the class currency,distance and timer.

7. Get the choice from the user and invoke the methods to perform the corresponding conversion and display the value.

PROGRAM

currencyconversion.java

```
package currencyconversion;
import java.util.*;
public class currency
{
double inr,usd;
double euro,yen;
Scanner in=new Scanner(System.in);
public void dollartorupee()
{
System.out.println("Enter dollars to convert into Rupees:");
usd=in.nextInt();
inr=usd*67;
System.out.println("Dollar ="+usd+"equal to INR="+inr);
}
```

```
public void rupeetodollar()
{
System.out.println("Enter Rupee to convert into Dollars:");
inr=in.nextInt();
usd=inr/67;
System.out.println("Rupee ="+inr+"equal to Dollars="+usd);
}
public void eurotorupee()
{
System.out.println("Enter euro to convert into Rupees:");
euro=in.nextInt();
inr=euro*79.50;
System.out.println("Euro ="+euro +"equal to INR="+inr);
}
public void rupeetoeuro()
{
System.out.println("Enter Rupees to convert into Euro:");
inr=in.nextInt();
euro=(inr/79.50);
System.out.println("Rupee ="+inr +"equal to Euro="+euro);
}
public void yentorupee()
{
System.out.println("Enter yen to convert into Rupees:");
yen=in.nextInt();
inr=yen*0.61;
System.out.println("YEN="+yen +"equal to INR="+inr);
}
public void rupeetoyen()
{
System.out.println("Enter Rupees to convert into Yen:");
inr=in.nextInt();
yen=(inr/0.61);
System.out.println("INR="+inr +"equal to YEN"+yen);
}
}
```

distance.java

```
package distanceconversion;
import java.util.*;
public class distance
{
double km,m,miles;
Scanner sc = new Scanner(System.in);
public void kmtom()
{
System.out.print("Enter in km ");
km=sc.nextDouble();
m=(km*1000);
System.out.println(km+"km" +"equal to"+m+"metres");
}
public void mtokm()
{
System.out.print("Enter in meter ");
m=sc.nextDouble();
km=(m/1000);
System.out.println(m+"m" +"equal to"+km+"kilometres");
}
public void milestokm()
{
System.out.print("Enter in miles");
miles=sc.nextDouble();
km=(miles*1.60934);
System.out.println(miles+"miles" +"equal to"+km+"kilometres");
}
public void kmtomiles()
{
System.out.print("Enter in km");
km=sc.nextDouble();
miles=(km*0.621371);
System.out.println(km+"km" +"equal to"+miles+"miles");
}
}
```

timer.java

```
package timeconversion;
import java.util.*;
public class timer
```

```
{
int hours,seconds,minutes;
int input;
Scanner sc = new Scanner(System.in);
public void secondstohours()
{
System.out.print("Enter the number of seconds: ");
input = sc.nextInt();
hours = input / 3600;
minutes = (input % 3600) / 60;
seconds = (input % 3600) % 60;
System.out.println("Hours: " + hours);
System.out.println("Minutes: " + minutes);
System.out.println("Seconds: " + seconds);
}
public void minutestohours()
{
System.out.print("Enter the number of minutes: ");
minutes=sc.nextInt();
hours=minutes/60;
minutes=minutes%60;
System.out.println("Hours: " + hours);
System.out.println("Minutes: " + minutes);
}
public void hourstominutes()
{
System.out.println("enter the no of hours");
hours=sc.nextInt();
minutes=(hours*60);
System.out.println("Minutes: " + minutes);
}
public void hourstoseconds()
{
System.out.println("enter the no of hours");
hours=sc.nextInt();
seconds=(hours*3600);
System.out.println("Minutes: " + seconds);
}
}
```

converter.java

```
import java.util.*;
import java.io.*;
import currencyconversion.*;
import distanceconversion.*;
import timeconversion.*;
class converter
{
public static void main(String args[])
{
Scanner s=new Scanner(System.in);
int choice,ch;
currency c=new currency();
distance d=new distance();
timer t=new timer();
do
{
System.out.println("1.dollar to rupee ");
System.out.println("2.rupee to dollar ");
System.out.println("3.Euro to rupee ");
System.out.println("4..rupee to Euro ");
System.out.println("5.Yen to rupee ");
System.out.println("6.Rupee to Yen ");
System.out.println("7.Meter to kilometer ");
System.out.println("8.kilometer to meter ");
System.out.println("9.Miles  to kilometer ");
System.out.println("10.kilometer to miles");
System.out.println("11.Hours to Minutes");
System.out.println("12.Hours to Seconds");
System.out.println("13.Seconds to Hours");
System.out.println("14.Minutes to Hours");
System.out.println("Enter ur choice");
choice=s.nextInt();
switch(choice)
{
case 1:
{
```

```
c.dollartorupee();
break;
}
case 2:
{
c.rupeetodollar();
break;
}
case 3:
{
c.eurotorupee();
break;
}
case 4:
{
c.rupeetoeuro();
break;
}
case 5:
{c.yentorupee();
break;}
case 6 :
{
c.rupeetoyen();
break;
}
case 7  :
{
d.mtokm();
break;
}
case 8  :
{
d.kmtom();
break;
}
case 9  :
{
d.milestokm();
```

```
break;
}
case 10  :
{
d.kmtomiles();
break;
}
case 11 :
{
t.hourstominutes();
break;
}
case 12 :
{
t.hourstoseconds();
break;
}
case 13 :
{
t.secondstohours();
break;
}
case 14 :
{
t.minutestohours();
break;
}}
System.out.println("Enter 0 to  quit and 1 to continue ");
ch=s.nextInt();
}while(ch==1);
}
}
```

OUTPUT

RESULT

Thus the java application to implement currency converter ,distance converter and time converter was implemented and executed successfully.

EX NO: 3 PROGRAM TO GENERATE PAYSLIP USING INHERITANCE

AIM

To develop a java application to generate pay slip for different category of employees using the concept of inheritance.

PROCEDURE

1. Create the class employee with name, Empid, address, mailid, mobileno as members.
2. Inherit the classes programmer, asstprofessor,associateprofessor and professor from
   employee class.
3. Add Basic Pay (BP) as the member of all the inherited classes.
4. Calculate DA as 97% of BP, HRA as 10% of BP, PF as 12% of BP, Staff club fund as 0.1% of
   BP.
5. Calculate gross salary and net salary.
6. Generate payslip for all categories of employees.

7. Create the objects for the inherited classes and invoke the necessary methods to display the
Payslip.

PROGRAM

```
import java.util.*;
class employee
{
 int empid;
 long mobile;
String name, address, mailid;
Scanner get = new Scanner(System.in);
void getdata()
{
```

```
 System.out.println("Enter Name of the Employee");
 name = get.nextLine();
 System.out.println("Enter Mail id");
 mailid = get.nextLine();
 System.out.println("Enter Address of the Employee:");
 address = get.nextLine();
 System.out.println("Enter employee id ");
 empid = get.nextInt();
 System.out.println("Enter Mobile Number");
 mobile = get.nextLong();
 }
 void displ ay()
 {
 System.out.println("Employee Name: "+name);
 System.out.println("Employee id : "+empid);
 System.out.println("Mail id : "+mailid);
 System.out.println("Address: "+address);
 System.out.println("Mobile Number: "+mobile);
 }
}
class programmer extends employee
{
double salary,bp,da,hra,pf,club,net,gross;
void getprogrammer()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateprog()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("***********************************************");
System.out.println("PAY SLIP FOR PROGRAMMER");
System.out.println("***********************************************");
```

```
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("PF:Rs"+pf);
System.out.println("HRA:Rs"+hra);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}
class asstprofessor extends employee
{
double salary,bp,da,hra,pf,club,net,gross;
void getasst()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateasst()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("***********************************************");
System.out.println("PAY SLIP FOR ASSISTANT PROFESSOR");
System.out.println("***********************************************");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
 }
}
class associateprofessor extends employee
{
```

```
double salary,bp,da,hra,pf,club,net,gross;
 void getassociate()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateassociate()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("************************************************");
System.out.println("PAY SLIP FOR ASSOCIATE PROFESSOR");
System.out.println("************************************************");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}
class professor extends employee
{
double salary,bp,da,hra,pf,club,net,gross;
void getprofessor()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateprofessor()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
```

```
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("***********************************************");
System.out.println("PAY SLIP FOR PROFESSOR");
System.out.println("***********************************************");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}
class salary
{
public static void main(String args[])
{
int choice,cont;
do
{
System.out.println("PAYROLL");
System.out.println(" 1.PROGRAMMER \t 2.ASSISTANT PROFESSOR \t
3.ASSOCIATE
PROFESSOR \t 4.PROFESSOR ");
Scanner c = new Scanner(System.in);
choice=c.nextInt();
switch(choice)
{
case 1:
{
programmer p=new programmer();
p.getdata();
p.getprogrammer();
p.display();
p.calculateprog();
break;
}
```

```
case 2:
{
asstprofessor asst=new asstprofessor();
asst.getdata();
asst.getasst();
asst.display();
asst.calculateasst();
break;
}
case 3:
{
associateprofessor asso=new associateprofessor();
asso.getdata();
asso.getassociate();
asso.display();
asso.calculateassociate();
break;
}
case 4:
{
professor prof=new professor();
prof.getdata();
prof.getprofessor();
prof.display();
prof.calculateprofessor();
break;
}
}
System.out.println("Do u want to continue 0 to quit and  1 to continue ");
cont=c.nextInt();
}while(cont==1);
}
}
```

OUTPUT

RESULT

Thus the java application to generate pay slip for different category of employees was implemented using inheritance and the program was executed successfully.

EX NO: 4 PROGRAM FOR STACK ADT USING INTERFACE

AIM

To design a java application to implement array implementation of stack using the concept of interface and exception handling.

PROCEDURE

1. Create the interface stackoperation with method declarations for push and pop.
2. Create the class astack which implements the interface and provides implementation for the methods push and pop.Also define the method for displaying the values stored in the stack.Handle the stack overflow and stack underflow condition .
3. Create the class teststack .Get the choice from the user for the operation to be performed . and also handle the exception that occur while performing the stack operation.
4. Create the object and invoke the method for push,pop,display based on the input from the user.

PROGRAM

```
import java.io.*;
interface stackoperation
{
 public void push(int i);
 public void pop();
}
class Astack implements stackoperation
{
int stack[]=new int[5];
int top=-1;
int i;
public void push(int item)
{
if(top>=4)
{
System.out.println("overflow");
}
else
{
top=top+1;
stack[top]=item;
System.out.println("item pushed"+stack[top]);
}
}
public void pop()
{
if(top<0)
System.out.println("underflow");
else
{
System.out.println("item popped"+stack[top]);
top=top-1;
}
}
public void display()
{
if(top<0)
```

```
System.out.println("No Element in stack");
else
{
for(i=0;i<=top;i++)
System.out.println("element:"+stack[i]);
}
}
}
class teststack
{
public static void main(String args[])throws IOException
{
int ch,c;
int i;
Astack s=new Astack();
DataInputStream in=new DataInputStream(System.in);
do
{
try
{
System.out.println("ARRAY STACK");
System.out.println("1.push  2.pop  3.display  4.exit");
System.out.println("enter ur choice:");
ch=Integer.parseInt(in.readLine());
switch(ch)
{
case 1:
System.out.println("enter the value to push:");
i=Integer.parseInt(in.readLine());
s.push(i);
break;
case 2:
s.pop();
break;
case 3:
System.out.println("the elements are:");
s.display();
break;
case 4:
```

```
break;
}
}
catch(IOException e)
{
System.out.println("io error");
}
System.out.println("Do u want to continue 0 to quit and  1 to continue ");
c=Integer.parseInt(in.readLine());
}while(c==1);
}
}
```

OUTPUT

RESULT

Thus the java application for stack operations has been implemented and executed successfully.

EX NO: 5 PROGRAM TO PERFORM STRING OPERATIONS USING ARRAYLIST

AIM

To write a java program to perform string operations using ArrayList for the following functions

a. Append - add at end
b. Insert – add at particular index
c. Search
d. List all string starts with given letter

PROCEDURE

1.Create the class arraylistexample. Create the object for the arraylist class.

2.Display the options to the user for performing string handling .

3.Use the function add() to append the string at the end and to insert the string at the particular index.

4. The function sort () is used to sort the elements in the array list.

5. The function indexof() is used to search whether the element is in the array list or not.

6.The function startsWith () is used to find whether the element starts with the specified character.

7.The function remove() is used to remove the element from the arraylist.

8.The function size() is used to determine the number of elements in the array list.

PROGRAM

```
import java.util.*;
import java.io.*;
public class arraylistexample
{
public static void main(String args[])throws IOException
{
ArrayList<String> obj = new ArrayList<String>();
DataInputStream in=new DataInputStream(System.in);
int c,ch;
int i,j;
String str,str1;
do
{
System.out.println("STRING MANIPULATION");
System.out.println("******************************");
System.out.println(" 1. Append at end \t 2.Insert at particular index \t 3.Search \t");
System.out.println( "4.List string that starting with letter \t");
System.out.println("5.Size \t 6.Remove \t 7.Sort\t 8.Display\t" );
System.out.println("Enter the choice ");
c=Integer.parseInt(in.readLine());
 switch(c)
 {
 case 1:
 {
 System.out.println("Enter the string ");
 str=in.readLine();
 obj.add(str);
 break;
 }
case 2:
{
```

```
System.out.println("Enter the string ");
str=in.readLine();
System.out.println("Specify the index/position to insert");
i=Integer.parseInt(in.readLine());
obj.add(i-1,str);
System.out.println("The array list has following elements:"+obj);
break;
}
case 3:
{
System.out.println("Enter the string to search ");
str=in.readLine();
j=obj.indexOf(str);
if(j==-1)
System.out.println("Element not found");
else
System.out.println("Index of:"+str+"is"+j);
break;
}
case 4:
{
System.out.println("Enter the character to List string that starts with specified
character");
str=in.readLine();
for(i=0;i<(obj.size()-1);i++)
{
str1=obj.get(i);
if(str1.startsWith(str))
{
System.out.println(str1);
}
}
break;
}
case 5:
{
System.out.println("Size of the list "+obj.size());
break;
}
```

```
case 6:
{
System.out.println("Enter the element to remove");
str=in.readLine();
if(obj.remove(str))
{
System.out.println("Element Removed"+str);
}
else
{
 System.out.println("Element not present");
}
 break;
}
 case 7:
{
Collections.sort(obj);
System.out.println("The array list has following elements:"+obj);
break;
}
case 8:
{
 System.out.println("The array list has following elements:"+obj);
 break;
}
}
 System.out.println("enter 0 to break and 1 to continue");
 ch=Integer.parseInt(in.readLine());
}while(ch==1);
}
}
```

OUTPUT

RESULT

Thus the java program to perform string operations using ArrayList has been implemented and executed successfully.

EX NO : 6 PROGRAM TO CALCULATE AREA USING ABSTRACT CLASS

AIM

To write a java program to calculate the area of rectangle,circle and triangle using the concept of abstract class.

PROCEDURE

1. Create an abstract class named shape that contains two integers and an empty method named
printarea().

2. Provide three classes named rectangle, triangle and circle such that each one of the classes
extends the class Shape.

3.Each of the inherited class from shape class should provide the implementation for the method
printarea().

4.Get the input and calculate the area of rectangle,circle and triangle .

5. In the shapeclass , create the objects for the three inherited classes and invoke the methods and
display the area values of the different shapes.

PROGRAM

```
import java.util.*;
abstract class shape
{
 int a,b;
 abstract public void printarea();
}
class rectangle extends shape
{
public int area_rect;
public void printarea()
{
  Scanner s=new Scanner(System.in);
  System.out.println("enter the length and breadth of rectangle");
  a=s.nextInt();
  b=s.nextInt();
  area_rect=a*b;
  System.out.println("Length of rectangle "+a +"breadth of rectangle "+b);
  System.out.println("The area ofrectangle is:"+area_rect);
 }
 }
```

```
class triangle extends shape
{
 double area_tri;
 public void printarea()
{
Scanner s=new Scanner(System.in);
System.out.println("enter the base and height of triangle");
a=s.nextInt();
b=s.nextInt();
System.out.println("Base of triangle "+a +"height of triangle  "+b);
area_tri=(0.5*a*b);
System.out.println("The area of triangle is:"+area_tri);
}
}
class circle extends shape
{
double area_circle;
public void printarea()
{
Scanner s=new Scanner(System.in);
System.out.println("enter the radius of circle");
a=s.nextInt();
 area_circle=(3.14*a*a);
 System.out.println("Radius of circle"+a);
 System.out.println("The area of circle is:"+area_circle);
 }
}
public class shapeclass
{
 public static void main(String[] args)
{
rectangle r=new rectangle();
r.printarea();
 triangle t=new triangle();
 t.printarea();
circle r1=new circle();
r1.printarea();
}
}
```

OUTPUT

RESULT

Thus a java program for calculate the area of rectangle,circle and triangle was

implemented and executed successfully.

EX NO : 7 PROGRAM TO IMPLEMENT USER DEFINED EXCEPTION HANDLING

AIM

To write a java program to implement user defined exception handling

PROCEDURE

1.Create a class which extends Exception class.
2.Create a constructor which receives the string as argument.
3.Get the Amount as input from the user.
4.If the amount is negative , the exception will be generated.
5.Using the exception handling mechanism , the thrown exception is handled by the catch
construct.
6.After the exception is handled , the string “invalid amount “ will be displayed.
7.If the amount is greater than 0 , the message “Amount Deposited “ will be displayed

(a)PROGRAM

```
import java.util.Scanner;
 class NegativeAmtException extends Exception
 {
 String msg;
 NegativeAmtException(String msg)
 {
```

```
        this.msg=msg;
 }
 public String toString()
 {
        return msg;
 }
}
public class userdefined
{
 public static void main(String[] args)
{
Scanner s=new Scanner(System.in);
System.out.print("Enter Amount:");
int a=s.nextInt();
try
{
if(a<0)
 {
throw new NegativeAmtException("Invalid Amount");
 }
 System.out.println("Amount Deposited");
 }
catch(NegativeAmtException e)
{
 System.out.println(e);
}
}
}
```

OUTPUT

(b) PROGRAM

```
class MyException extends Exception{
String str1;
MyException(String str2)
 {
str1=str2;
  }
  public String toString()
 {
return ("MyException Occurred: "+str1) ;
 }
}
class example
{
 public static void main(String args[])
{
try
{
System.out.println("Starting of try block");
throw new MyException("This is My error Message");
}
catch(MyException exp)
{
System.out.println("Catch Block") ;
System.out.println(exp) ;
}
}}
```

OUTPUT

RESULT

Thus a java program to implement user defined exception handling has been implemented and executed successfully.

EX NO :8 PROGRAM FOR DISPLAYING FILE INFORMATION

AIM

To write a java program that reads a file name from the user, displays information about whether the file exists, whether the file is readable, or writable, the type of file and the length of the file in bytes.

PROCEDURE

1.Create a class filedemo. Get the file name from the user .
2.Use the file functions and display the information about the file.
3.getName() displays the name of the file.
4.getPath() diplays the path name of the file.
5.getParent () -This method returns the pathname string of this abstract pathname’s parent, or
null if this pathname does not name a parent directory.
6.exists() – Checks whether the file exists or not.
7. canRead()-This method is basically a check if the file can be read.
8. canWrite()-verifies whether the application can write to the file.
9. isDirectory() – displays whether it is a directory or not.
10. isFile() – displays whether it is a file or not.
11. lastmodified() – displays the last modified information.
12.length()- displays the size of the file.
13. delete() – deletes the file
14.Invoke the predefined functions abd display the iformation about the file.

PROGRAM

```
import java.io.*;
import java.util.*;
class filedemo
{
public static void main(String args[])
{
String filename;
Scanner s=new Scanner(System.in);
System.out.println("Enter the file name ");
filename=s.nextLine();
File f1=new File(filename);
System.out.println("*****************");
System.out.println("FILE INFORMATION");
System.out.println("*****************");
System.out.println("NAME OF THE FILE "+f1.getName());
System.out.println("PATH OF THE FILE "+f1.getPath());
System.out.println("PARENT"+f1.getParent());
if(f1.exists())
System.out.println("THE FILE EXISTS ");
else
System.out.println("THE FILE DOES NOT ExISTS ");
if(f1.canRead())
System.out.println("THE FILE CAN BE READ ");
else
System.out.println("THE FILE CANNOT BE READ  ");
if(f1.canWrite())
System.out.println("WRITE OPERATION IS PERMITTED");
```

```
else
System.out.println("WRITE OPERATION IS NOT PERMITTED");
if(f1.isDirectory())
System.out.println("IT IS A DIRECTORY ");
else
System.out.println("NOT A DIRECTORY");
if(f1.isFile())
System.out.println("IT IS A FILE ");
else
System.out.println("NOT A FILE");
System.out.println("File last modified "+ f1.lastModified());
System.out.println("LENGTH OF THE FILE  "+f1.length());
System.out.println("FILE DELETED "+f1.delete());
}
}
```

OUTPUT

RESULT

Thus a java program to display file information has been implemented and executed successfully.

EX NO :9 PROGRAM TO IMPLEMENT MULTITHREADED APPLICATION

AIM

To write a java program that implements a multi-threaded application .

PROCEDURE

1.Create a class even which implements first thread that computes .the square of the number .

2. run() method implements the code to be executed when thread gets executed.

3.Create a class odd which implements second thread that computes the cube of the number.

4.Create a third thread that generates random number.If the random number is even , it displays
the square of the number.If the random number generated is odd , it displays the cube of the
given number .

5.The Multithreading is performed and the task switched between multiple threads.

6.The sleep () method makes the thread to suspend for the specified time.

PROGRAM

```
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 A extends Thread
{
public void run()
{
int num = 0;
Random r = new Random();
```

```
try
{
for (int i = 0; i < 5; 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 multithreadprog
{
public static void main(String[] args)
{
A a = new A();
a.start();
}
}
```

OUTPUT

RESULT

Thus a java program for multi-threaded application has been implemented and executed successfully.

EX NO: 10 PROGRAM TO FIND THE MAXIMUM VALUE FROM THE GIVEN

TYPE OF ELEMENTS USING GENERICS

AIM

To write a java program to find the maximum value from the given type of elements using a generic function.

PROCEDURE

1.Create a class Myclass to implement generic class and generic methods.
2.Get the set of the values belonging to specific data type.
3.Create the objects of the class to hold integer,character and double values.
4.Create the method to compare the values and find the maximum value stored in the array.
5.Invoke the method with integer, character or double values . The output will be displayed
based on the data type passed to the method.

PROGRAM

```
class MyClass<T extends Comparable<T>>
{
T[] vals;
MyClass(T[] o)
{
vals = o;
}
public T min()
{
T v = vals[0];
for(int i=1; i < vals.length; i++)
if(vals[i].compareTo(v) < 0)
v = vals[i];
```

```
return v;
}
public T max()
{
T v = vals[0];
for(int i=1; i < vals.length;i++)
if(vals[i].compareTo(v) > 0)
v = vals[i];
return v;
}
}
class gendemo
{
public static void main(String args[])
{
int i;
Integer inums[]={10,2,5,4,6,1};
Character chs[]={'v','p','s','a','n','h'};
Double d[]={20.2,45.4,71.6,88.3,54.6,10.4};
MyClass<Integer> iob = new MyClass<Integer>(inums);
MyClass<Character> cob = new MyClass<Character>(chs);
MyClass<Double>dob = new MyClass<Double>(d);
System.out.println("Max value in inums: " + iob.max());
System.out.println("Min value in inums: " + iob.min());
System.out.println("Max value in chs: " + cob.max());
System.out.println("Min value in chs: " + cob.min());
System.out.println("Max value in chs: " + dob.max());
System.out.println("Min value in chs: " + dob.min());
}
}
```

OUTPUT

RESULT

Thus a java program to find the maximum value from the given type of elements has been implemented using generics and executed successfully.

EX NO: 11 PROGRAM TO DESIGN A CALCULATOR USING EVENT DRIVEN

PROGRAMMING PARADIGM OF JAVA

AIM

To design a calculator using event driven programming paradigm of Java with the following options

a) Decimal Manipulations
b) Scientific Manipulations

PROCEDURE

1. import the swing packages and awt packages.
2. Create the class scientificcalculator that implements action listener.
3. Create the container and add controls for digits , scientific calculations and decimal Manipulations.
4. The different layouts can be used to lay the controls.
5. When the user presses the control , the event is generated and handled .
6. The corresponding decimal , numeric and scientific calculations are performed.

PROGRAM

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.event.*;
public class ScientificCalculator extends JFrame implements ActionListener
{
JTextField tfield;
double temp, temp1, result, a;
static double m1, m2;
int k = 1, x = 0, y = 0, z = 0;
char ch;
JButton b1, b2, b3, b4, b5, b6, b7, b8, b9, zero, clr, pow2, pow3, exp,
fac, plus, min, div, log, rec, mul, eq, addSub, dot, mr, mc, mp,
mm, sqrt, sin, cos, tan;
Container cont;
JPanel textPanel, buttonpanel;
ScientificCalculator()
{
cont = getContentPane();
cont.setLayout(new BorderLayout());
JPanel textpanel = new JPanel();
tfield = new JTextField(25);
tfield.setHorizontalAlignment(SwingConstants.RIGHT);
tfield.addKeyListener(new KeyAdapter() {
public void keyTyped(KeyEvent keyevent) {
char c = keyevent.getKeyChar();
if (c >= '0' && c <= '9') {
}
else
{
```

```
keyevent.consume();
}
}
});
textpanel.add(tfield);
buttonpanel = new JPanel();
buttonpanel.setLayout(new GridLayout(8, 4, 2, 2));
boolean t = true;
mr = new JButton("MR");
buttonpanel.add(mr);
mr.addActionListener(this);
mc = new JButton("MC");
buttonpanel.add(mc);
mc.addActionListener(this);
mp = new JButton("M+");
buttonpanel.add(mp);
mp.addActionListener(this);
mm = new JButton("M-");
buttonpanel.add(mm);
mm.addActionListener(this);
b1 = new JButton("1");
buttonpanel.add(b1);
b1.addActionListener(this);
b2 = new JButton("2");
buttonpanel.add(b2);
b2.addActionListener(this);
b3 = new JButton("3");
buttonpanel.add(b3);
b3.addActionListener(this);
b4 = new JButton("4");
buttonpanel.add(b4);
b4.addActionListener(this);
b5 = new JButton("5");
buttonpanel.add(b5);
b5.addActionListener(this);
b6 = new JButton("6");
buttonpanel.add(b6);
b6.addActionListener(this);
b7 = new JButton("7");
```

```
    buttonpanel.add(b7);
    b7.addActionListener(this);
    b8 = new JButton("8");
    buttonpanel.add(b8);
    b8.addActionListener(this);
    b9 = new JButton("9");
    buttonpanel.add(b9);
    b9.addActionListener(this);
    zero = new JButton("0");
    buttonpanel.add(zero);
    zero.addActionListener(this);
    plus = new JButton("+");
    buttonpanel.add(plus);
    plus.addActionListener(this);
    min = new JButton("-");
    buttonpanel.add(min);
    min.addActionListener(this);
    mul = new JButton("*");
    buttonpanel.add(mul);
    mul.addActionListener(this);
    div = new JButton("/");
    div.addActionListener(this);
    buttonpanel.add(div);
    addSub = new JButton("+/-");
    buttonpanel.add(addSub);
    addSub.addActionListener(this);
    dot = new JButton(".");
    buttonpanel.add(dot);
    dot.addActionListener(this);
    eq = new JButton("=");
    buttonpanel.add(eq);
    eq.addActionListener(this);
    rec = new JButton("1/x");
    buttonpanel.add(rec);
    rec.addActionListener(this);
    sqrt = new JButton("Sqrt");
    buttonpanel.add(sqrt);
    sqrt.addActionListener(this);
    log = new JButton("log");
```

```
buttonpanel.add(log);
log.addActionListener(this);
sin = new JButton("SIN");
buttonpanel.add(sin);
sin.addActionListener(this);
cos = new JButton("COS");
buttonpanel.add(cos);
cos.addActionListener(this);
tan = new JButton("TAN");
buttonpanel.add(tan);
tan.addActionListener(this);
pow2 = new JButton("x^2");
buttonpanel.add(pow2);
pow2.addActionListener(this);
pow3 = new JButton("x^3");
buttonpanel.add(pow3);
pow3.addActionListener(this);
exp = new JButton("Exp");
exp.addActionListener(this);
buttonpanel.add(exp);
fac = new JButton("n!");
fac.addActionListener(this);
buttonpanel.add(fac);
clr = new JButton("AC");
buttonpanel.add(clr);
clr.addActionListener(this);
cont.add("Center", buttonpanel);
cont.add("North", textpanel);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
public void actionPerformed(ActionEvent e)
{
String s = e.getActionCommand();
if (s.equals("1"))
{
if (z == 0)
{
tfield.setText(tfield.getText() + "1");
}
```

```
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "1");
z = 0;
}
}
if (s.equals("2")) {
if (z == 0) {
tfield.setText(tfield.getText() + "2");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "2");
z = 0;
}
}
if (s.equals("3")) {
if (z == 0) {
tfield.setText(tfield.getText() + "3");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "3");
z = 0;
}
}
if (s.equals("4")) {
if (z == 0) {
tfield.setText(tfield.getText() + "4");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "4");
z = 0;
}
```

```
}
if (s.equals("5")) {
if (z == 0) {
tfield.setText(tfield.getText() + "5");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "5");
z = 0;
}
}
if (s.equals("6")) {
if (z == 0) {
tfield.setText(tfield.getText() + "6");
}
else
 {
tfield.setText("");
tfield.setText(tfield.getText() + "6");
z = 0;
}
}
if (s.equals("7")) {
if (z == 0) {
tfield.setText(tfield.getText() + "7");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "7");
z = 0;
}
}
if (s.equals("8")) {
if (z == 0) {
tfield.setText(tfield.getText() + "8");
}
else
```

```
{
tfield.setText("");
tfield.setText(tfield.getText() + "8");
z = 0;
}
}
if (s.equals("9")) {
if (z == 0) {
tfield.setText(tfield.getText() + "9");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "9");
z = 0;
}
}
if (s.equals("0"))
{
if (z == 0) {
tfield.setText(tfield.getText() + "0");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "0");
z = 0;
}
}
if (s.equals("AC")) {
tfield.setText("");
x = 0;
y = 0;
z = 0;
}
if (s.equals("log"))
{
if (tfield.getText().equals("")) {
tfield.setText("");
```

```
}
else
 {
a = Math.log(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("1/x")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = 1 / Double.parseDouble(tfield.getText());
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("Exp")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.exp(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("x^2")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.pow(Double.parseDouble(tfield.getText()), 2);
tfield.setText("");
tfield.setText(tfield.getText() + a);
```

```
}
}
if (s.equals("x^3")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.pow(Double.parseDouble(tfield.getText()), 3);
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("+/-")) {
if (x == 0) {
tfield.setText("-" + tfield.getText());
x = 1;
}
else
{
tfield.setText(tfield.getText());
}
}
if (s.equals(".")) {
if (y == 0) {
tfield.setText(tfield.getText() + ".");
y = 1;
}
else
{
tfield.setText(tfield.getText());
}
}
if (s.equals("+"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
temp = 0;
```

```
ch = '+';
}
else
{
temp = Double.parseDouble(tfield.getText());
tfield.setText("");
ch = '+';
y = 0;
x = 0;
}
tfield.requestFocus();
}
if (s.equals("-"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
temp = 0;
ch = '-';
}
 else
 {
x = 0;
y = 0;
temp = Double.parseDouble(tfield.getText());
tfield.setText("");
ch = '-';
}
tfield.requestFocus();
}
if (s.equals("/")) {
if (tfield.getText().equals(""))
{
tfield.setText("");
temp = 1;
ch = '/';
}
else
{
```

```
x = 0;
y = 0;
temp = Double.parseDouble(tfield.getText());
ch = '/';
tfield.setText("");
}
tfield.requestFocus();
}
if (s.equals("*")) {
if (tfield.getText().equals(""))
{
tfield.setText("");
temp = 1;
ch = '*';
}
else
{
x = 0;
y = 0;
temp = Double.parseDouble(tfield.getText());
ch = '*';
tfield.setText("");
}
tfield.requestFocus();
}
if (s.equals("MC"))
{
m1 = 0;
tfield.setText("");
}
if (s.equals("MR"))
{
tfield.setText("");
tfield.setText(tfield.getText() + m1);
}
if (s.equals("M+"))
{
if (k == 1) {
m1 = Double.parseDouble(tfield.getText());
```

```
k++;
}
else
{
m1 += Double.parseDouble(tfield.getText());
tfield.setText("" + m1);
}
}
if (s.equals("M-"))
{
if (k == 1) {
m1 = Double.parseDouble(tfield.getText());
k++;
}
else
 {
m1 -= Double.parseDouble(tfield.getText());
tfield.setText("" + m1);
}
}
if (s.equals("Sqrt"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
a = Math.sqrt(Double.parseDouble(tfield.getText()));
tfield.setText("");
field.setText(tfield.getText() + a);
}
}
if (s.equals("SIN"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
```

```
else
{
a = Math.sin(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("COS"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
a = Math.cos(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("TAN")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.tan(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("="))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
```

```
temp1 = Double.parseDouble(tfield.getText());
switch (ch)
{
case '+':
result = temp + temp1;
break;
case '-':
result = temp - temp1;
break;
case '/':
result = temp / temp1;
break;
case '*':
result = temp * temp1;
break;
}
tfield.setText("");
tfield.setText(tfield.getText() + result);
z = 1;
}
}
if (s.equals("n!"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
a = fact(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
tfield.requestFocus();
}
double fact(double x)
{
int er = 0;
```

```
if (x < 0)
{
er = 20;
return 0;
}
double i, s = 1;
for (i = 2; i <= x; i += 1.0)
s *= i;
return s;
}
public static void main(String args[])
{
try
{
UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel"
);
}
catch (Exception e)
{
}
ScientificCalculator f = new ScientificCalculator();
f.setTitle("ScientificCalculator");
f.pack();
f.setVisible(true);
}
}
```

OUTPUT

RESULT

Thus the java programs for scientific calculator has been implemented and executed successfully.

EX NO :12 MINI PROJECT FOR LIBRARY MANAGEMENT SYSTEM

CODING

Logon.java

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

public class Logon extends JFrame implements ActionListener
{
int fl=1;
JPanel pLog = new JPanel();
JLabel lbUser, lbPass;
JTextField txtUser;
JPasswordField txtPass;
JButton btnOk, btnCancel;
Connection con;
public String user;
public Logon ()
{
super ("Library Management System.");
setSize (275, 300);
addWindowListener (new WindowAdapter ()
{
public void windowClosing (WindowEvent we) {
setVisible (false);
```

```
dispose();
System.exit (0);
}
}
);
pLog.setLayout (null);
lbUser = new JLabel ("Username:");
lbUser.setForeground (Color.black);
lbUser.setBounds (20, 15, 75, 25);
lbPass = new JLabel ("Password:");
lbPass.setForeground (Color.BLACK);
lbPass.setBounds (20, 50, 75, 25);
txtUser = new JTextField ();
txtUser.setBounds (100, 15, 150, 25);
txtPass = new JPasswordField ();
txtPass.setBounds (100, 50, 150, 25);
//Setting the Form's Buttons.
btnOk = new JButton ("OK");
btnOk.setBounds (20, 90, 100, 25);
btnOk.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (150, 90, 100, 25);
btnCancel.addActionListener (this);
pLog.add (lbUser);
pLog.add (lbPass);
pLog.add (txtUser);
pLog.add (txtPass);
pLog.add (btnOk);
pLog.add (btnCancel);
getContentPane().add (pLog);
//Opening the Database.
try
{
Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
String loc = "jdbc:odbc:temp1";
con = DriverManager.getConnection (loc);
}
catch (ClassNotFoundException cnf)  {
JOptionPane.showMessageDialog (null, "Driver not Loaded...");
```

```
System.exit (0);
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "Unable to Connect to Database...");
System.exit (0);
}
//Showing The Logon Form.
setVisible (true);
}
public void actionPerformed (ActionEvent ae)
{
Object obj = ae.getSource();
if (obj == btnOk)
{
String password = new String (txtPass.getPassword());
if (txtUser.getText().equals (""))
{
JOptionPane.showMessageDialog (this, "Provide Username to Logon.");
txtUser.requestFocus();
}
else if (password.equals (""))
{
txtPass.requestFocus();
JOptionPane.showMessageDialog (null,"Provide Password to Logon.");
}
else
{
String pass;
boolean verify = false;
if(fl==1)
{
if(txtUser.getText().equals("Admin")&&password.equals("admin"))
{
verify=true;
new LibrarySystem(1,1,con);
setVisible(false);
dispose();
}
}
```

```
else
{
String tablename=null;
if(fl==2) tablename="Clerks";
else if(fl==3)tablename="Members";
try {  //SELECT Query to Retrieved the Record.
String query = "SELECT * FROM " + tablename + " WHERE id = " +
Integer.parseInt(txtUser.getText());
Statement st = con.createStatement ();
ResultSet rs = st.executeQuery (query);
rs.next();
user = rs.getString ("id");
pass = rs.getString ("Password");
if (txtUser.getText().equals (user) && password.equals (pass))
{
verify = true;
new LibrarySystem (fl,Integer.parseInt(txtUser.getText()), con);
setVisible (false);
dispose();
}
else
{
verify = false;
txtUser.setText ("");
txtPass.setText ("");
txtUser.requestFocus ();
}
}
catch (Exception sqlex)
 {
if (verify == false)
{
txtUser.setText ("");
txtPass.setText ("");
txtUser.requestFocus ();
}
}
}
}
```

```
}
else if (obj == btnCancel)
{
setVisible (false);
dispose();
System.exit (0);
}
}
public static void main(String args[])
{
Logon start=new Logon();
}
}
class FrmSplash extends JWindow implements Runnable
{
Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
public void run()
{
setSize(275,300);
setVisible(true);
}
}
```

AddBCat.java

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

public class AddBCat extends JInternalFrame implements ActionListener
{
JPanel pNew = new JPanel();
JLabel lbUser;
JTextField txtUser;
JButton btnOk, btnCancel;

private Statement st;
public AddBCat (Connection con)
```

```
{
super ("New Book Category", false, true, false, true);
setSize (280, 175);
lbUser = new JLabel ("Category:");
lbUser.setForeground (Color.black);
lbUser.setBounds (20, 20, 100, 25);
txtUser = new JTextField ();
txtUser.setBounds (100, 20, 150, 25);
btnOk = new JButton ("OK");
btnOk.setBounds (20, 100, 100, 25);
btnOk.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (150, 100, 100, 25);
btnCancel.addActionListener (this);
pNew.setLayout (null);
pNew.add (lbUser);
pNew.add (txtUser);
pNew.add (btnOk);
pNew.add (btnCancel);
getContentPane().add (pNew);
try
{
st = con.createStatement ();
}
catch (SQLException sqlex)
{
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading the Form.");
dispose ();
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae)
{
Object obj = ae.getSource();
if (obj == btnOk)
{
if (txtUser.getText().equals ("")) {
txtUser.requestFocus();
JOptionPane.showMessageDialog (this, "Category not Provided.");
```

```
}
else
{
try
{
String id= txtUser.getText();
String q = "SELECT * FROM BCat ";
ResultSet rs = st.executeQuery (q);
int fl=0;
while(rs.next())
{
String memberNo = rs.getString ("Cat");
if(id.equals(memberNo))
{
JOptionPane.showMessageDialog(this,"Already existing Category");
txtUser.setText("");
txtUser.requestFocus();
fl=1;
break;
}
}
rs.close();
if(fl==0){
q = "INSERT INTO BCat " + "VALUES ('" + txtUser.getText() + "')";
int result = st.executeUpdate (q);
if (result == 1) {
JOptionPane.showMessageDialog (this, "New Category Created.");
txtUser.setText ("");
txtUser.requestFocus ();
}
else
{
JOptionPane.showMessageDialog (this, "Problem while Creating. ");
txtUser.setText ("");
txtUser.requestFocus ();
}
}
}
catch (SQLException sqlex)
```

```
{
JOptionPane.showMessageDialog (this, "Problem while Creating excep.");
}
}
}
if (obj == btnCancel) {
setVisible (false);
dispose();
}
}
}
```

AddBook.java

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

public class AddBook extends JInternalFrame implements ActionListener,
FocusListener
{
JPanel pBook = new JPanel ();
JLabel lbBookId, lbBookName, lbBookAuthor, lbBookRef, lbBookCategory;
JTextField txtBookId, txtBookName, txtBookAuthor;
JComboBox cboBookCategory;
JButton btnOk, btnCancel;
JRadioButton rby,rbn;
ButtonGroup bg;
String[] cn =new String[100];
Statement st;
long id = 0;
int i,j,ref=0;
public AddBook (Connection con)
{
super ("Add New Book", false ,true, true, true);
setSize (325, 250);
lbBookId = new JLabel ("Book Id:");
lbBookId.setForeground (Color.black);
```

```
lbBookId.setBounds (15, 15, 100, 20);
lbBookName = new JLabel ("Book Name:");
lbBookName.setForeground (Color.black);
lbBookName.setBounds (15, 45, 100, 20);
lbBookAuthor = new JLabel ("Book Author:");
lbBookAuthor.setForeground (Color.black);
lbBookAuthor.setBounds (15, 75, 100, 20);
lbBookRef = new JLabel ("Reference:");
lbBookRef.setForeground (Color.black);
lbBookRef.setBounds (15, 105, 100, 20);
lbBookCategory = new JLabel ("Book Category:");
lbBookCategory.setForeground (Color.black);
lbBookCategory.setBounds (15, 135, 100, 20);
txtBookId = new JTextField ();
txtBookId.setHorizontalAlignment (JTextField.RIGHT);
txtBookId.addFocusListener (this);
txtBookId.setBounds (120, 15, 175, 25);
txtBookName = new JTextField ();
txtBookName.setBounds (120, 45, 175, 25);
txtBookAuthor = new JTextField ();
txtBookAuthor.setBounds (120, 75, 175, 25);
rby=new JRadioButton("yes");
rby.addActionListener(this);
rby.setBounds(120,105,60,25);
rbn=new JRadioButton("no");
rbn.addActionListener(this);
rbn.setBounds(180,105,60,25);
bg = new ButtonGroup();
bg.add(rby);
bg.add(rbn);
rbn.setSelected(true);
cboBookCategory = new JComboBox();
cboBookCategory.setBounds (120, 135, 175, 25);
btnOk = new JButton ("OK");
btnOk.setBounds (50, 175, 100, 25);
btnOk.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (170, 175, 100, 25);
btnCancel.addActionListener (this);
```

```
txtBookId.addKeyListener (new KeyAdapter ()
{
public void keyTyped (KeyEvent ke)
{
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE)))
{
getToolkit().beep ();
ke.consume ();
}
}
}
);
txtBookAuthor.addKeyListener (new KeyAdapter ()
{
public void keyTyped (KeyEvent ke)
{
char c = ke.getKeyChar ();
if (! ((Character.isLetter (c)) || (c == KeyEvent.VK_BACK_SPACE)||(c ==
KeyEvent.VK_SPACE)))
{
getToolkit().beep ();
ke.consume ();
}
}
}
);
pBook.setLayout (null);
pBook.add (lbBookId);
pBook.add (lbBookName);
pBook.add (lbBookAuthor);
pBook.add (lbBookRef);
pBook.add (lbBookCategory);
pBook.add (txtBookId);
pBook.add (txtBookName);
pBook.add (txtBookAuthor);
pBook.add (rby);
pBook.add (rbn);
pBook.add (cboBookCategory);
```

```
pBook.add (btnOk);
pBook.add (btnCancel);
getContentPane().add (pBook, BorderLayout.CENTER);
try {
i=0;
st = con.createStatement ();
ResultSet rs=st.executeQuery("Select * from BCat");
while(rs.next())
{
cn[i]=rs.getString(1);
i++;
}
for(j=0;j<i;j++)
{
cboBookCategory.addItem(cn[j]);
}
cboBookCategory.addActionListener(this);
cboBookCategory.setSelectedItem(cn[0]);
rs.close();
}
catch (SQLException sqlex)
{
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading Form.");
dispose ();
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae)
{
Object obj = ae.getSource();
if (obj == btnOk)
{
if (txtBookId.getText().equals (""))
{
JOptionPane.showMessageDialog (this, "Book's Id not Provided.");
txtBookId.requestFocus ();
}
else if (txtBookName.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Book's Name not Provided.");
```

```
txtBookName.requestFocus ();
}
else if (txtBookAuthor.getText().equals (""))
{
JOptionPane.showMessageDialog (this, "Book's Author Name not Provided.");
txtBookAuthor.requestFocus ();
}
else
{
try {
int x = 0;
String s8 = x+"/"+x+"/"+x ;

//INSERT Query to Add Book Record in Table.
/* String q = "INSERT INTO Books " +"VALUES (" + id + ", '" + txtBookName.getText() +
"', '" + txtBookAuthor.getText() + "', " + ref + ", '" + cboBookCategory.getSelectedItem()
+ "' ,"+ 0 + ", '" + s8 + "', '" + s8 + ")"; */

int result = st.executeUpdate ("Insert into Books values("+ id +",'" +
txtBookName.getText() +"','" + txtBookAuthor.getText() +"', " + ref + ", '" +
cboBookCategory.getSelectedItem().toString() +"', " + 0 + ", '"+ s8 +"' ,'"+ s8 + "')");
//Running Query.
if (result == 1) {
//If Query Successful.
JOptionPane.showMessageDialog (this, "Record has been Saved.");
txtClear ();                    //Clearing the TextFields.
}
else
 {
//If Query Failed.
OptionPane.showMessageDialog (this, "Problem while Saving the Record.");
}
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (this, "Problem while Saving the Record Excep.");
}
}
}
if (obj == btnCancel) {     /If Cancel Button Pressed Unload the From.
```

```
setVisible (false);
dispose();

}
if(obj==rby)
{
ref=1;
}
else if(obj==rbn)
{
ref=0;
}
}
public void focusGained (FocusEvent fe) { }
public void focusLost (FocusEvent fe) {
if (txtBookId.getText().equals ("")) {
}
else {
id = Integer.parseInt (txtBookId.getText ());
long bookNo;
boolean found = false;
try {
String q = "SELECT * FROM Books WHERE BId = " + id + "";
ResultSet rs = st.executeQuery (q);
rs.next ();
bookNo = rs.getLong ("BId");
if (bookNo == id) {
found = true;
txtClear ();
JOptionPane.showMessageDialog (this, id + " is already assigned.");
}
else {
found = false;
}
}
catch (SQLException sqlex)
{
}
}
```

```
}
private void txtClear () {
txtBookId.setText ("");
txtBookName.setText ("");
txtBookAuthor.setText ("");
cboBookCategory.setSelectedIndex(0);
txtBookId.requestFocus ();
}
}
```

AddMCat.java

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

public class AddMCat extends JInternalFrame implements ActionListener {
JPanel pNew = new JPanel();
JLabel lbUser,lbDate,lbBooks;
JTextField txtUser,txtDate,txtBooks;
JButton btnOk, btnCancel;
private Statement st;
public AddMCat (Connection con) {
super ("New Member Category", false, true, false, true);
setSize (280, 200);
lbUser = new JLabel ("Category:");
lbUser.setForeground (Color.black);
lbUser.setBounds (20, 20, 100, 25);
lbDate = new JLabel ("Days Issued:");
lbDate.setForeground (Color.black);
lbDate.setBounds (20, 55, 100, 25);
lbBooks= new JLabel ("No. of Books");
lbBooks.setForeground (Color.black);
lbBooks.setBounds (20, 90, 100, 25);
txtUser = new JTextField ();
txtUser.setBounds (100, 20, 150, 25);
txtDate = new JTextField ();
txtDate.setBounds (100, 55, 150, 25);
txtDate.addKeyListener (new KeyAdapter () {
```

```
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);
txtBooks = new JTextField();
txtBooks.setBounds(100,90,150,25);
txtBooks.addKeyListener (new KeyAdapter () {
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);
btnOk = new JButton ("OK");
btnOk.setBounds (20, 123, 100, 25);
btnOk.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (150, 123, 100, 25);
btnCancel.addActionListener (this);
pNew.setLayout (null);
pNew.add (lbUser);
pNew.add (lbDate);
pNew.add (lbBooks);
pNew.add (txtUser);
pNew.add (txtDate);
pNew.add (txtBooks);
pNew.add (btnOk);
pNew.add (btnCancel);
getContentPane().add (pNew);
try {
st = con.createStatement ();
```

```
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading the Form.");
dispose ();
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == btnOk) {
if (txtUser.getText().equals ("")) {
txtUser.requestFocus();
JOptionPane.showMessageDialog (this, "Username not Provided.");
}
else {
try {
String id= txtUser.getText();
String q = "SELECT CName FROM MeCat ";
ResultSet rs = st.executeQuery (q);
int fl=0;
while(rs.next())
{
String memberNo = rs.getString ("CName");
if(id.equals(memberNo))
{
JOptionPane.showMessageDialog(this,"Already existing Category");
txtUser.setText("");
txtUser.requestFocus();
fl=1;
break;
}
}
rs.close();
int num=0;
try{
rs= st.executeQuery("Select * From MeCat");
while(rs.next())
{
num++;
```

```
}
num++;
rs.close();
}
catch(Exception e)
{
JOptionPane.showMessageDialog (this, "Problem while Creating excep1.");
}
if(fl==0){
int result = st.executeUpdate ("Insert into MeCat Values(" + num + ", '" +
txtUser.getText() + "' ," + Integer.parseInt(txtBooks.getText()) + " , " +
Integer.parseInt(txtDate.getText())+ " )" );//Running Query.
if (result == 1) {
JOptionPane.showMessageDialog (this, "New Category Created.");
txtUser.setText ("");
txtUser.requestFocus ();
}
else {
JOptionPane.showMessageDialog (this, "Problem while Creating. ");
txtUser.setText ("");
txtUser.requestFocus ();
}
}
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (this, "Problem while Creating excep.");
}
}

}
if (obj == btnCancel) {
setVisible (false);
dispose();
}
}
}
```

Addmember.java

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

public class AddMember extends JInternalFrame implements ActionListener,
FocusListener {
JPanel pMember = new JPanel ();
JLabel lbMemberId, lbMemberName, lbMemberpwd, lbEntryDate,lbCategory;
JTextField txtMemberId, txtMemberName, txtMemberpwd,txtMemberdate;
JButton btnOk, btnCancel;
JComboBox cboMemCategory;
Statement st;
long id = 0;
String[] cn= new String[100];
int id1,im,iy,vd,vm,vy,i;
public AddMember (Connection con) {
super ("Add New Member", false, true, false, true);
setSize (355, 250);
lbMemberId = new JLabel ("Member Id:");
lbMemberId.setForeground (Color.black);
lbMemberId.setBounds (15, 15, 100, 20);
lbMemberName = new JLabel ("Member Name:");
lbMemberName.setForeground (Color.black);
lbMemberName.setBounds (15, 45, 100, 20);
lbMemberpwd = new JLabel ("Member Pwd:");
lbMemberpwd.setForeground (Color.black);
lbMemberpwd.setBounds (15, 75, 110, 20);
lbEntryDate = new JLabel ("Entry Date:");
lbEntryDate.setForeground (Color.black);
lbEntryDate.setBounds (15, 105, 100, 20);
lbCategory = new JLabel ("Category:");
lbCategory.setForeground(Color.BLACK);
lbCategory.setBounds(15,135,100,20);

txtMemberId = new JTextField ();
txtMemberId.setHorizontalAlignment (JTextField.RIGHT);
```

```
txtMemberId.addFocusListener (this);
txtMemberId.setBounds (125, 15, 205, 25);
txtMemberName = new JTextField ();
txtMemberName.setBounds (125, 45, 205, 25);
txtMemberpwd = new JTextField ();
txtMemberpwd.setBounds (125, 75, 205, 25);
txtMemberdate=new JTextField();
txtMemberdate.setBounds(125,105,205,25);
txtMemberdate.setEditable(false);
cboMemCategory= new JComboBox();
cboMemCategory.setBounds(125,135,100,20);
GregorianCalendar gcal=new GregorianCalendar();
id= gcal.get(Calendar.DATE);
im=(int)gcal.get(Calendar.MONTH)+1;
iy=gcal.get(Calendar.YEAR);
String idate=id+"/"+im+"/"+iy;
txtMemberdate.setText(idate);
btnOk = new JButton ("OK");
btnOk.setBounds (30, 165, 125, 25);
btnOk.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (190, 165, 125, 25);
btnCancel.addActionListener (this);
txtMemberId.addKeyListener (new KeyAdapter () {
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);
txtMemberName.addKeyListener(new KeyAdapter() {
public void keyTyped(KeyEvent ke) {
char c = ke.getKeyChar();
if (! ((Character.isLetter(c)) || (c == KeyEvent.VK_BACK_SPACE)||(c ==
KeyEvent.VK_SPACE))) {
getToolkit().beep();
```

```
ke.consume();
}
}
}
 );
pMember.setLayout (null);
pMember.add (lbMemberId);
pMember.add (lbMemberName);
pMember.add (lbMemberpwd);
pMember.add (lbEntryDate);
pMember.add (txtMemberId);
pMember.add (txtMemberName);
pMember.add (txtMemberpwd);
pMember.add(txtMemberdate);
pMember.add (btnOk);
pMember.add (btnCancel);
pMember.add (lbCategory);
pMember.add (cboMemCategory);
getContentPane().add (pMember, BorderLayout.CENTER);
int j;
try {
i=0;
st = con.createStatement ();
ResultSet rs=st.executeQuery("Select * from MeCat");
while(rs.next())
{
cn[i]=rs.getString(2);
i++;
}
for(j=0;j<i;j++)
{
cboMemCategory.addItem(cn[j]);
}
cboMemCategory.addActionListener(this);
cboMemCategory.setSelectedItem(cn[0]);
rs.close();
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading Form.");
```

```
dispose ();
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == btnOk) {
if (txtMemberId.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Member's Id not Provided.");
txtMemberId.requestFocus ();
}
else if (txtMemberName.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Member's Name not Provided.");
txtMemberName.requestFocus ();
}
else if (txtMemberpwd.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Member's Password not Provided.");
txtMemberpwd.requestFocus ();
}
else {
try {
int mtype=cboMemCategory.getSelectedIndex()+1;
String q = "INSERT INTO Members" + " VALUES (" + id + ", '" +
txtMemberpwd.getText() + "', '" + txtMemberName.getText() + "', '" +
txtMemberdate.getText() + "',"+ 0 + "," + 0 + "," + mtype + ")";
int result = st.executeUpdate (q);
if (result == 1) {
JOptionPane.showMessageDialog (this, "Record has been Saved.");
txtClear ();
}
else {
JOptionPane.showMessageDialog (this, "Problem while Saving the Record.");
}
}
catch (SQLException sqlex) {JOptionPane.showMessageDialog(this,"Error!!"); }
}
}
if (obj == btnCancel) {
setVisible (false);
```

```
dispose();
}
}
public void focusGained (FocusEvent fe) { }
public void focusLost (FocusEvent fe) {
if (txtMemberId.getText().equals ("")) {
}
else {
id = Integer.parseInt (txtMemberId.getText ());
long memberNo;
boolean found = false;
try {
String q = "SELECT * FROM Members WHERE id = " + id + "";
ResultSet rs = st.executeQuery (q);
rs.next ();
memberNo = rs.getLong ("id");
if (memberNo == id) {
found = true;
txtClear ();
JOptionPane.showMessageDialog (this, id + " is already assigned.");
}
else
{
found = false;
}
}
catch (SQLException sqlex) { }
}
}
private void txtClear () {
txtMemberId.setText ("");
txtMemberName.setText ("");
txtMemberpwd.setText ("");
txtMemberId.requestFocus ();
}
}
```

Dates.java

```
public class Dates
{
int m,d,y;
public Dates(int month, int day, int year)
{
m=month;
d=day;
y=year;
}

public int getMonth()
{ return m; }

public int getDay()
{ return d; }

public int getYear()
{ return y; }

public void setMonth(int month)
{ m=month;  }

public void setDay(int day)
{ d=day;    }

public void setYear(int year)
{ y=year;   }

public String toString()
{
String s = m + "/" + d + "/" + y;
return s;
}

public long toLong()
{
long days=0;

switch(m)
```

```
{
case 12: days+=30;
case 11: days+=31;
case 10: days+=30;
case 9:  days+=31;
case 8:  days+=31;
case 7:  days+=30;
case 6:  days+=31;
case 5:  days+=30;
case 4:  days+=31;
case 3:  days+= isLeapYear(y) ? 29 : 28;
case 2:  days+=31;
case 1:  days+=d-1;
}

if(y!=1900) {
int inc=(1900-y)/Math.abs(1900-y);
for(int i=y; i!=1900; i+=inc)
days += (isLeapYear(i) ? 366 : 365);
}

return days;
}

private boolean isLeapYear(int y)
{
 if((y%100)==0) return (y%400)==0;
 else return (y%4)==0;
 }

public long getDifference(Dates date)
{ return Math.abs(toLong()-date.toLong()); }

}
```

DeleteBook.java

```
import java.awt.*;
import java.awt.event.*;
```

```
import javax.swing.*;
import java.sql.*;
import java.util.*;
public class DeleteBook extends JInternalFrame implements ActionListener,
FocusListener {
JPanel pBook = new JPanel ();
JLabel lbBookId, lbBookName, lbBookAuthor;
JTextField txtBookId, txtBookName, txtBookAuthor;
JButton btnDel, btnCancel;
Statement st;
ResultSet rs;
private long id = 0,bisued;
public DeleteBook (Connection con) {
super ("Delete Book", false, true, false, true);
setSize (325, 250);
lbBookId = new JLabel ("Book Id:");
lbBookId.setForeground (Color.black);
lbBookId.setBounds (15, 15, 100, 20);
lbBookName = new JLabel ("Book Name:");
lbBookName.setForeground (Color.black);
lbBookName.setBounds (15, 45, 100, 20);
lbBookAuthor = new JLabel ("Book Author:");
lbBookAuthor.setForeground (Color.black);
lbBookAuthor.setBounds (15, 75, 100, 20);
txtBookId = new JTextField ();
txtBookId.setHorizontalAlignment (JTextField.RIGHT);
txtBookId.addFocusListener (this);
txtBookId.setBounds (120, 15, 175, 25);
txtBookName = new JTextField ();
txtBookName.setEnabled (false);
txtBookName.setBounds (120, 45, 175, 25);
txtBookAuthor = new JTextField ();
txtBookAuthor.setEnabled (false);
txtBookAuthor.setBounds (120, 75, 175, 25);
btnDel = new JButton ("Delete Book");
btnDel.setBounds (25, 175, 125, 25);
btnDel.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (165, 175, 125, 25);
```

```
btnCancel.addActionListener (this);
txtBookId.addKeyListener (new KeyAdapter () {
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);

pBook.setLayout (null);
pBook.add (lbBookId);
pBook.add (lbBookName);
pBook.add (lbBookAuthor);
pBook.add (txtBookId);
pBook.add (txtBookName);
pBook.add (txtBookAuthor);
pBook.add (btnDel);
pBook.add (btnCancel);
getContentPane().add (pBook, BorderLayout.CENTER);
try {
st = con.createStatement ();
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading the Form.");
dispose ();
}
setVisible (true);
}

public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == btnDel) {
if (txtBookId.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Book's Id not Provided.");
txtBookId.requestFocus ();
}
```

```
else if(bisued!=0)
{
txtClear();
JOptionPane.showMessageDialog(this,"Book held by a member");
}
else
{

int reply = JOptionPane.showConfirmDialog (this, "Are you really want to Delete\nthe "
+ txtBookName.getText () + " Record?","LibrarySystem - Delete Book",
JOptionPane.YES_NO_OPTION, JOptionPane.PLAIN_MESSAGE);

if (reply == JOptionPane.YES_OPTION) {
try {
String q = "DELETE FROM Books WHERE BId = " + id + "";
txtClear ();
JOptionPane.showMessageDialog (this, "Book Deleted.");
ResultSet rs = st.executeQuery (q);
}
catch (SQLException sqlex) { }
}

else if (reply == JOptionPane.NO_OPTION) { }
}
}
if (obj == btnCancel) {
setVisible (false);
dispose();
}
}
public void focusGained (FocusEvent fe) { }
public void focusLost (FocusEvent fe) {
if (txtBookId.getText().equals ("")) {
}
else {
id = Integer.parseInt (txtBookId.getText ());
long bookNo;
boolean found = false;
try {
```

```
String q = "SELECT * FROM Books WHERE BId = " + id + "";
ResultSet rs = st.executeQuery (q);
rs.next ();
bookNo = rs.getLong ("BId");
bisued=rs.getLong("Mid");
if (bookNo == id) {
found = true;
txtBookId.setText ("" + id);
txtBookName.setText ("" + rs.getString ("BName"));
txtBookAuthor.setText ("" + rs.getString ("BAuthor"));
}
else {
found = false;
}
}
catch (SQLException sqlex) {
if (found == false) {
txtClear ();
JOptionPane.showMessageDialog (this, "Record not Found.");
}
}
}
}
private void txtClear () {
txtBookId.setText ("");
txtBookName.setText ("");
txtBookAuthor.setText ("");
txtBookId.requestFocus();
}
}
```

DeleteMember.java

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

public class DeleteMember extends JInternalFrame implements ActionListener,
```

```
FocusListener {
private JPanel pMember = new JPanel ();
private JLabel lbMemberId, lbMemberName,lbMemberCat;
private JTextField txtMemberId, txtMemberName,txtCat;
private JButton btnDel, btnCancel;
private int due;
private Statement st;              //Statement for Getting the Required Table.
private ResultSet rs;              //For Getting the Records From Table.
private long id = 0,heldBooks;        //To Hold the MemberId.
//Constructor of Class.
public DeleteMember (Connection con) {
//super (Title, Resizable, Closable, Maximizable, Iconifiable)
super ("Delete Member", false, true, false, true);
setSize (350, 222);
//Setting the Form's Labels.
lbMemberId = new JLabel ("Member Id:");
lbMemberId.setForeground (Color.black);
lbMemberId.setBounds (15, 15, 100, 20);
lbMemberName = new JLabel ("Member Name:");
lbMemberName.setForeground (Color.black);
lbMemberName.setBounds (15, 45, 100, 20);
lbMemberCat = new JLabel ("Category");
lbMemberCat.setForeground (Color.black);
lbMemberCat.setBounds (15, 75, 110, 20);
txtMemberId = new JTextField ();
txtMemberId .setHorizontalAlignment (JTextField.RIGHT);
txtMemberId.addFocusListener (this);
txtMemberId .setBounds (125, 15, 200, 25);
txtMemberName = new JTextField ();
txtMemberName.setEnabled (false);
txtMemberName.setBounds (125, 45, 200, 25);
txtCat = new JTextField ();
txtCat.setEnabled (false);
txtCat.setBounds (125, 75, 200, 25);
btnDel = new JButton ("Delete Member");
btnDel.setBounds (30, 145, 125, 25);
btnDel.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (185, 145, 125, 25);
```

```
btnCancel.addActionListener (this);
//Registering the KeyListener to Restrict user to type only Numeric in Numeric Boxes.
txtMemberId.addKeyListener (new KeyAdapter () {
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);
//Adding All the Controls in Panel.
pMember.setLayout (null);
pMember.add (lbMemberId);
pMember.add (lbMemberName);
pMember.add(lbMemberCat);
pMember.add(txtCat);
pMember.add (txtMemberId);
pMember.add (txtMemberName);
pMember.add (btnDel);
pMember.add (btnCancel);
//Adding Panel to Form.
getContentPane().add (pMember, BorderLayout.CENTER);
try {
st = con.createStatement ();    //Creating Statement Object.
}
catch (SQLException sqlex) {                //If Problem then Show the User a Message.
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading the Form.");
dispose ();                                 //Closing the Form.
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == btnDel) {          //If Delete Button Pressed.
if (txtMemberId.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Member's Id not Provided.");
txtMemberId.requestFocus ();
```

```
}
else if(heldBooks!=0)
{
JOptionPane.showMessageDialog(this,"Member Holding Books..Can't Delete");
txtClear();
}
else if(due!=0)
{
JOptionPane.showMessageDialog(this,"Member Holding Books..Can't Delete");
txtClear();
}
else
{
int reply = JOptionPane.showConfirmDialog (this, "Do you really want to Delete\nthe "
+ txtMemberName.getText () + " Record?","LibrarySystem - Delete Member",
JOptionPane.YES_NO_OPTION, JOptionPane.PLAIN_MESSAGE);

//Check the User Selection.
if (reply == JOptionPane.YES_OPTION) {                    //If User's Choice Yes then.
try {  //DELETE Query to Delete the Record from Table.
String q = "DELETE FROM Members WHERE id = " + id + "";
txtClear ();                              //Clearing the TextFields.
JOptionPane.showMessageDialog (this, "Record has been Deleted.");
ResultSet rs = st.executeQuery (q);       //Executing the Query.
}
catch (SQLException sqlex) {System.out.println("problem"); }
}
//If User's Choice No then Do Nothing Return to Program.
else if (reply == JOptionPane.NO_OPTION) { }
}
}
if (obj == btnCancel) {     //If Cancel Button Pressed Unload the From.
setVisible (false);
dispose();
}
}

//OverRidding the FocusListener Class Function.
public void focusGained (FocusEvent fe) { }
```

```
public void focusLost (FocusEvent fe) {
if (txtMemberId.getText().equals ("")) {    //If TextField is Empty.
}
else {
id = Integer.parseInt (txtMemberId.getText ());//Converting String to Numeric.
long memberNo,memtype;                  //Use for Comparing the Member's Id.
boolean found = false;                  //To Confirm the Member's Id Existance.
try {  //SELECT Query to Retrieved the Record.
String q = "SELECT * FROM Members WHERE id = " + id + "";
ResultSet rs = st.executeQuery (q);       //Executing the Query.
rs.next ();                     //Moving towards the Record.
memberNo = rs.getLong ("id");       //Storing the Record.
heldBooks=rs.getLong("Bcnt");
due=rs.getInt(6);
memtype=rs.getLong("Mcat");
if (memberNo == id) {      //If Record Found then Display Records.
found = true;
txtMemberId.setText ("" + id);
txtMemberName.setText ("" + rs.getString ("MName"));
ResultSet rs1=st.executeQuery("Select * From MeCat where Mcat="+memtype+"");
rs1.next();
txtCat.setText(""+rs1.getString("CName"));
}
else {
found = false;
}
}
catch (SQLException sqlex) {
if (found == false) {
txtClear ();              //Clearing the TextFields.
JOptionPane.showMessageDialog (this, "Record not Found.");
}
}
}
}
//Function Use to Clear All the TextFields of Form.
private void txtClear () {
txtMemberId.setText ("");
txtMemberName.setText ("");
```

```
txtCat.setText("");
txtMemberId.requestFocus ();
}
}
```

IssueBook.java

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

public class IssueBook extends JInternalFrame implements ActionListener,
FocusListener {
private JPanel pBook = new JPanel ();
private JLabel lbBookId, lbBookName, lbBookAuthor, lbBookCategory, lbMemberId,
lbMemberName,lbDate1,lbDate2;
private JTextField txtBookId, txtBookName, txtBookAuthor, txtBookCategory,
txtMemberId, txtMemberName,txtDate1,txtDate2;
private JButton btnOk, btnCancel;
private Statement st;
private long id = 0;
private int memberId = 0;
private int id1,im,iy,vd,vm,vy;
private String idate,vdate;
public IssueBook (Connection con) {
super ("Issue Book", false, true, false, true);
setSize (325, 340);
lbBookId = new JLabel ("Book Id:");
lbBookId.setForeground (Color.black);
lbBookId.setBounds (15, 15, 100, 20);
lbBookName = new JLabel ("Book Name:");
lbBookName.setForeground (Color.black);
lbBookName.setBounds (15, 45, 100, 20);
lbBookAuthor = new JLabel ("Book Author:");
lbBookAuthor.setForeground (Color.black);
lbBookAuthor.setBounds (15, 75, 100, 20);
```

```
lbBookCategory = new JLabel ("Book Category:");
lbBookCategory.setForeground (Color.black);
lbBookCategory.setBounds (15, 105, 100, 20);
lbMemberId = new JLabel ("Member Id:");
lbMemberId.setForeground (Color.black);
lbMemberId.setBounds (15, 135, 100, 20);
lbMemberName = new JLabel ("Member Name:");
lbMemberName.setForeground (Color.black);
lbMemberName.setBounds (15, 165, 100, 20);
lbDate1 = new JLabel ("Issue Date:");
lbDate1.setForeground (Color.black);
lbDate1.setBounds (15, 195, 100, 20);
lbDate2 = new JLabel ("Return Date:");
lbDate2.setForeground (Color.black);
lbDate2.setBounds (15, 225, 100, 20);
txtBookId = new JTextField ();
txtBookId.setHorizontalAlignment (JTextField.RIGHT);
txtBookId.addFocusListener (this);
txtBookId.setBounds (120, 15, 175, 25);
txtBookName = new JTextField ();
txtBookName.setEnabled (false);
txtBookName.setBounds (120, 45, 175, 25);
txtBookAuthor = new JTextField ();
txtBookAuthor.setEnabled (false);
txtBookAuthor.setBounds (120, 75, 175, 25);
txtBookCategory = new JTextField ();
txtBookCategory.setEnabled (false);
txtBookCategory.setBounds (120, 105, 175, 25);
txtMemberId = new JTextField ();
txtMemberId.setHorizontalAlignment (JTextField.RIGHT);
txtMemberId.addFocusListener (this);
txtMemberId.setBounds (120, 135, 175, 25);
txtMemberName = new JTextField ();
txtMemberName.setEnabled (false);
txtMemberName.setBounds (120, 165, 175, 25);
txtDate1 = new JTextField ();
txtDate1.setEnabled (false);
txtDate1.setBounds (120, 195, 175, 25);
txtDate1.setEditable(false);
```

```
txtDate2 = new JTextField ();
txtDate2.setEnabled (false);
txtDate2.setBounds (120, 225, 175, 25);
txtDate2.setEditable(false);
btnOk = new JButton ("OK");
btnOk.setBounds (50, 260, 100, 25);
btnOk.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (170, 260, 100, 25);
btnCancel.addActionListener (this);
txtBookId.addKeyListener (new KeyAdapter () {
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);

txtMemberId.addKeyListener (new KeyAdapter () {
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);

pBook.setLayout (null);
pBook.add (lbBookId);
pBook.add (lbBookName);
pBook.add (lbBookAuthor);
pBook.add (lbBookCategory);
pBook.add (lbMemberId);
pBook.add (lbMemberName);
```

```
pBook.add (txtBookId);
pBook.add (txtBookName);
pBook.add (txtBookAuthor);
pBook.add (txtBookCategory);
pBook.add (txtMemberId);
pBook.add (txtMemberName);
pBook.add (btnOk);
pBook.add (btnCancel);
pBook.add (txtDate1);
pBook.add (txtDate2);
pBook.add (lbDate1);
pBook.add (lbDate2);
getContentPane().add (pBook, BorderLayout.CENTER);
try {
st = con.createStatement ();
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading Form.");
dispose ();
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == btnOk) {
if (txtBookId.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Book's Id not Provided.");
txtBookId.requestFocus ();
}
else if (txtMemberId.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Member's Id not Provided.");
txtMemberId.requestFocus ();
}
else {
try  {
int i1= Integer.parseInt(txtMemberId.getText());
ResultSet rs = st.executeQuery("Select * from Members where id="+ i1 +"");
rs.next();
int bc=rs.getInt("Bcnt");
```

```
bc++;
int bid1=Integer.parseInt(txtBookId.getText());
int result = st.executeUpdate ("Update Members SET Bcnt="+ bc +" WHERE id="+ i1
+"");
if (result == 1) {
txtClear ();
}
else {
txtClear ();
JOptionPane.showMessageDialog (this, "Problem while Saving the Record.");
}
System.out.println("came 1");
result = st.executeUpdate("Update Books SET Mid= "+ i1 + ", BIssue = '"+ idate +"',
BReturn = '"+vdate+"' where Bid="+bid1+"");
System.out.println("came 2");
if (result == 1) {
txtClear ();
JOptionPane.showMessageDialog (this, "Record has been Saved.");
}
else {
txtClear ();
JOptionPane.showMessageDialog (this, "Problem while Saving the Record.");
}
}
catch (SQLException sqlex) {JOptionPane.showMessageDialog (this, "Problem"); }
}
}
if (obj == btnCancel) {
setVisible (false);
dispose();
}
}
public void focusGained (FocusEvent fe) { }
public void focusLost (FocusEvent fe) {
Object obj = fe.getSource ();

if (obj == txtBookId) {
if (txtBookId.getText().equals ("")) {
}
```

```
else {
id = Integer.parseInt (txtBookId.getText ());
long bookNo;
boolean found = false;
try {
String q = "SELECT * FROM Books WHERE BId = " + id + "";
ResultSet rs = st.executeQuery (q);
rs.next ();
bookNo = rs.getLong ("BId");
int mid=rs.getInt("Mid");
int bref=rs.getInt("BRef");
if(bref==1)
{
txtClear();
JOptionPane.showMessageDialog (this, "Ref Book Can't Be Issued.");
}
if(mid!=0)
{
txtClear();
JOptionPane.showMessageDialog(this,"Book Already Issued");
}
if (bookNo == id) {
found = true;
txtBookId.setText ("" + id);
txtBookName.setText ("" + rs.getString ("BName"));
txtBookAuthor.setText ("" + rs.getString ("BAuthor"));
txtBookCategory.setText ("" + rs.getString ("BCat"));
}
else {
found = false;
}
}
catch (SQLException sqlex) {
if (found == false) {
txtBookId.requestFocus ();
txtBookId.setText ("");
txtBookName.setText ("");
txtBookAuthor.setText ("");
txtBookCategory.setText ("");
```

```
JOptionPane.showMessageDialog (this, "Record not Found.");
}
}
}
}
else if (obj == txtMemberId) {
if (txtMemberId.getText().equals ("")) {
}
else {
memberId = Integer.parseInt (txtMemberId.getText ());
int memberNo,memberDays,memberBooks,memberCat,heldBooks;
boolean find = false;
try {
String q = "SELECT * FROM Members WHERE id = " + memberId + "";
ResultSet rs = st.executeQuery (q);
rs.next ();
memberNo = rs.getInt ("id");
if (memberNo == memberId) {
find = true;
memberCat=rs.getInt("MCat");
heldBooks=rs.getInt("Bcnt");
txtMemberName.setText ("" + rs.getString ("MName"));
rs.close();
ResultSet rs1= st.executeQuery("Select * from Mecat where MCat = " + memberCat +
"" );
rs1.next();
memberBooks=rs1.getInt("Blmt");
memberDays=rs1.getInt("Dlmt");
if(heldBooks==memberBooks)
{
txtClear();
JOptionPane.showMessageDialog (this, "Book Limit Reached");
dispose();
}

GregorianCalendar gcal=new GregorianCalendar();
 id1= gcal.get(Calendar.DATE);
 im=(int)gcal.get(Calendar.MONTH)+1;
 iy=gcal.get(Calendar.YEAR);
```

```
 vd=id1+memberDays;
 vm=im;
 vy=iy;
 String xx,yy,zz;
 if(id1<10) {
 xx="0"+id1;
 }
 else
{
 xx = ""+id1;
 }
if(im<10) {
yy="0"+im;
}
else
{
 yy = ""+im;
 }
 idate=xx+"/"+yy+"/"+iy;
 while(vd>31) {
 if(im==1||im==3||im==5||im==7||im==8||im==10||im==12)
{
if(vd>31){
im=im+1;
 vd=vd-31;
  if(im>12){
  im=im-12;
  iy=iy+1;
  }}}

 if(im==4||im==6||im==9||im==11){
 if(vd>30){
 im=im+1;
vd=vd-30;
if(im>12){
im=im-12;
 iy=iy+1;}
 }}
if(im==2){
```

```
if(vd>28){
im=im+1;
vd=vd-28;
if(im>12){
im=im-12;
iy=iy+1;
}}}
}
vdate = vd+"/"+im+"/"+iy;
txtMemberId.setText ("" + memberId);
txtDate1.setText(idate);
txtDate2.setText(vdate);
}
else {
find = false;
}
}
catch (SQLException sqlex) {
if (find == false) {
txtClear ();
JOptionPane.showMessageDialog (this, "Record not Found.");
}
}
}
}
}
}
private void txtClear () {
txtBookId.setText ("");
txtBookName.setText ("");
txtBookAuthor.setText ("");
txtBookCategory.setText ("");
txtMemberId.setText ("");
txtMemberName.setText ("");
txtBookId.requestFocus ();
}
}
```

LibrarySystem.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;
import java.util.*;
import java.text.*;
import java.io.*;

public class LibrarySystem extends JFrame implements ActionListener
{
private JDesktopPane desktop = new JDesktopPane ();
JMenuBar bar;
JMenu mnuFile, mnuEdit;
JMenuItem newBook,newMember, printBook,  printIssueBook;
JMenuItem issueBook, returnBook, delBook, findBook;
private     JToolBar toolBar;
private     JButton btnNewBook,  btnIssue, btnReturn, btnPrintIssue,
btnDelBook,btnFindBook;
private JPanel statusBar = new JPanel ();
Connection con;
Statement st;
String userName;
public LibrarySystem (int type,int user, Connection conn)
{
super ("Library Management System.");
setIconImage (getToolkit().getImage ("Images/Warehouse.png"));
setSize (700, 550);
setLocation((Toolkit.getDefaultToolkit().getScreenSize().width  - getWidth()) / 2,
(Toolkit.getDefaultToolkit().getScreenSize().height - getHeight()) / 2);
addWindowListener (new WindowAdapter () {
public void windowClosing (WindowEvent we) {
//quitApp ();
}
}
);
bar = new JMenuBar ();
setJMenuBar (bar);
mnuFile = new JMenu ("File");
mnuFile.setMnemonic ((int)'E');
```

```
mnuEdit = new JMenu ("Edit");
mnuEdit.setMnemonic ((int)'E');
newBook = new JMenuItem ("Add New Book");
newBook.setAccelerator (KeyStroke.getKeyStroke(KeyEvent.VK_N,
Event.CTRL_MASK));
newBook.setMnemonic ((int)'N');
newBook.addActionListener (this);
newMember = new JMenuItem ("Add New Member");
newMember.setAccelerator (KeyStroke.getKeyStroke(KeyEvent.VK_M,
Event.CTRL_MASK));
newMember.setMnemonic ((int)'M');
newMember.addActionListener (this);
issueBook = new JMenuItem ("Issue Book");
issueBook.setAccelerator (KeyStroke.getKeyStroke(KeyEvent.VK_I,
Event.CTRL_MASK));
issueBook.setMnemonic ((int)'I');
issueBook.addActionListener (this);
returnBook = new JMenuItem ("Return Book");
returnBook.setAccelerator (KeyStroke.getKeyStroke(KeyEvent.VK_R,
Event.CTRL_MASK));
returnBook.setMnemonic ((int)'R');
returnBook.addActionListener (this);
delBook = new JMenuItem ("Delete Book");
delBook.setAccelerator (KeyStroke.getKeyStroke(KeyEvent.VK_D,
Event.CTRL_MASK));
delBook.setMnemonic ((int)'D');
delBook.addActionListener (this);
findBook = new JMenuItem ("Search Book");
findBook.setAccelerator (KeyStroke.getKeyStroke(KeyEvent.VK_F,
Event.CTRL_MASK));
findBook.setMnemonic ((int)'F');
findBook.addActionListener (this);
mnuFile.add (newBook);
mnuFile.add (newMember);
mnuEdit.add (issueBook);
mnuEdit.add (returnBook);
mnuEdit.addSeparator ();
mnuEdit.add (delBook);
mnuEdit.addSeparator ();
```

```
mnuEdit.add (findBook);
bar.add (mnuFile);
bar.add (mnuEdit);
btnNewBook = new JButton (new ImageIcon ("Images/NotePad.gif"));
btnNewBook.setToolTipText ("Add New Book");
btnIssue = new JButton (new ImageIcon ("Images/Film.gif"));
btnIssue.setToolTipText ("Issue Book");
btnReturn = new JButton (new ImageIcon ("Images/Backup.gif"));
btnReturn.setToolTipText ("Return Book");
btnDelBook = new JButton (new ImageIcon ("Images/Recycle.gif"));
btnDelBook.setToolTipText ("Delete Book");
btnFindBook = new JButton (new ImageIcon ("Images/Mirror.gif"));
btnFindBook.setToolTipText ("Search Book");
btnFindBook.addActionListener (this);
toolBar = new JToolBar ();
toolBar.add (btnNewBook);
toolBar.addSeparator ();
toolBar.add (btnIssue);
toolBar.add (btnReturn);
toolBar.addSeparator ();
toolBar.add (btnDelBook);
toolBar.addSeparator ();
toolBar.add (btnFindBook);
if(type==1)
userName="Admin";
else if(type==2)
{
}
else if(type==3)
{
}

//Setting the Contents of Programs.
getContentPane().add (toolBar, BorderLayout.NORTH);
getContentPane().add (desktop, BorderLayout.CENTER);
getContentPane().add (statusBar, BorderLayout.SOUTH);
//Getting the Database.
con = conn;
setVisible (true);
```

```
}
public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == newBook ) {


boolean b = openChildWindow ("Add New Book");
if (b == false) {
AddBook adBook = new AddBook (con);
desktop.add (adBook);
adBook.show ();
}
}
else if (obj == newMember )
 {
boolean b = openChildWindow ("Add New Member");
if (b == false) {
AddMember adMember = new AddMember (con);
desktop.add (adMember);
adMember.show ();
}
}
else if (obj == issueBook )
 {
boolean b = openChildWindow ("Issue Book");
if (b == false) {
IssueBook isBook = new IssueBook (con);
desktop.add (isBook);
isBook.show ();
}
}
else if (obj == returnBook)
{
boolean b = openChildWindow ("Return Book");
if (b == false) {
ReturnBook rtBook = new ReturnBook (con);
desktop.add (rtBook);
rtBook.show ();
}
```

```
}
else if (obj == delBook)
{
boolean b = openChildWindow ("Delete Book");
if (b == false)
{
DeleteBook dlBook = new DeleteBook (con);
desktop.add (dlBook);
dlBook.show ();
}
}
else if (obj == findBook )
{
boolean b = openChildWindow ("Search Books");
if (b == false) {
SearchBook srBook = new SearchBook (con);
desktop.add (srBook);
srBook.show ();
}
}
}
private boolean openChildWindow (String title)
{
JInternalFrame[] childs = desktop.getAllFrames ();
for (int i = 0; i < childs.length; i++) {
if (childs[i].getTitle().equalsIgnoreCase (title))
{
childs[i].show ();
return true;
}
}
return false;
}
}
```

ReturnBook.java

```
import java.io.*;
import java.awt.*;
```

```
import java.awt.event.*;
import javax.swing.*;
import java.util.Date;
import java.util.Calendar;
import java.util.*;
import java.text.SimpleDateFormat;
import java.sql.*;
public class ReturnBook extends JInternalFrame implements ActionListener,
FocusListener {
private JPanel pBook = new JPanel ();
private JLabel lbBookId, lbBookName,lbIssued;
private JTextField txtBookId, txtBookName, txtIssued;
private String urdate;
private JButton btnReturn, btnCancel;
private int id1,im,iy,vd,vm,vy,due;
private Statement st;
private ResultSet rs;
private long id = 0;
private int mid,bc;
public ReturnBook (Connection con) {
super ("Return Book", false, true, false, true);
setSize (325, 250);
lbBookId = new JLabel ("Book Id:");
lbBookId.setForeground (Color.black);
lbBookId.setBounds (15, 15, 100, 20);
lbBookName = new JLabel ("Book Name:");
lbBookName.setForeground (Color.black);
lbBookName.setBounds (15, 45, 100, 20);
lbIssued = new JLabel ("Book Issued To:");
lbIssued.setForeground (Color.black);
lbIssued.setBounds (15, 75, 100, 20);
txtBookId = new JTextField ();
txtBookId.setHorizontalAlignment (JTextField.RIGHT);
txtBookId.addFocusListener (this);
txtBookId.setBounds (120, 15, 175, 25);
txtBookName = new JTextField ();
txtBookName.setEnabled (false);
txtBookName.setBounds (120, 45, 175, 25);
txtIssued = new JTextField ();
```

```
txtIssued.setEnabled (false);
txtIssued.setBounds (120, 75, 175, 25);
btnReturn = new JButton ("Return Book");
btnReturn.setBounds (25, 175, 125, 25);
btnReturn.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (165, 175, 125, 25);
btnCancel.addActionListener (this);
txtBookId.addKeyListener (new KeyAdapter () {
public void keyTyped (KeyEvent ke) {
char c = ke.getKeyChar ();
if (! ((Character.isDigit (c)) || (c == KeyEvent.VK_BACK_SPACE))) {
getToolkit().beep ();
ke.consume ();
}
}
}
);
pBook.setLayout (null);
pBook.add (lbBookId);
pBook.add (lbBookName);
pBook.add (lbIssued);
pBook.add (txtBookId);
pBook.add (txtBookName);
pBook.add (txtIssued);
pBook.add (btnReturn);
pBook.add (btnCancel);
getContentPane().add (pBook, BorderLayout.CENTER);
try {
st = con.createStatement ();
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading the Form.");
dispose ();
}
GregorianCalendar gcal=new GregorianCalendar();
 id1= gcal.get(Calendar.DATE);
 im=(int)gcal.get(Calendar.MONTH)+1;
 iy=gcal.get(Calendar.YEAR);
```

```
 String xx,yy,zz;
 if(id1<10) {
 xx="0"+id1;
} else {
 xx = ""+id1;
 }
if(im<10) {
yy="0"+im;
}
 else
{
  yy = ""+im;
 }
 urdate=xx+"/"+yy+"/"+iy;
setVisible (true);
}
public void actionPerformed (ActionEvent ae)
{
Object obj = ae.getSource();
if (obj == btnReturn) {
if (txtBookId.getText().equals (""))
{
JOptionPane.showMessageDialog (this, "Book's Id not Provided.");
txtBookId.requestFocus ();
}
else {
try {
int rd,rm,ry,urd,urm,ury,x;
long v,v1,fine;
Dates d1,d2;
bc--;
id = Integer.parseInt (txtBookId.getText ());
ResultSet rs = st.executeQuery ("select * from Books WHERE BId ="+id+"");
//Executing the Query.
rs.next();
String ard=rs.getString("BReturn");
System.out.println("came here 1");
rs.close();
String sr=urdate;
```

```
StringTokenizer st2 = new StringTokenizer(sr,"/");
urd=Integer.parseInt(st2.nextToken());
urm=Integer.parseInt(st2.nextToken());
ury=Integer.parseInt(st2.nextToken());
d2= new Dates(urm,urd,ury);
StringTokenizer st1 = new StringTokenizer(ard,"/");
rd=Integer.parseInt(st1.nextToken());
rm=Integer.parseInt(st1.nextToken());
ry=Integer.parseInt(st1.nextToken());
d1=new Dates(rm,rd,ry);
v = d1.toLong();
v1 = d2.toLong();
fine=v1-v;
if(fine<=0)
fine=0;
else
{
int reply = JOptionPane.showConfirmDialog (this, "Will you pay the Fine of
Rs."+fine+"now","FinePay", JOptionPane.YES_NO_OPTION,
JOptionPane.PLAIN_MESSAGE);
if (reply == JOptionPane.YES_OPTION)
{}
else if (reply == JOptionPane.NO_OPTION)
{
due+=fine;
}
}
x=st.executeUpdate("Update Books Set Mid ="+0+" WHERE Bid ="+id+"");
x=st.executeUpdate("Update Members Set Bcnt ="+bc+", Mbdues="+due+" WHERE id
="+mid+"");
JOptionPane.showMessageDialog (this, "Book Returned");
txtClear();
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (this, "Problem");
}
}
}
if (obj == btnCancel) {
setVisible (false);
```

```
dispose();
}
}
public void focusGained (FocusEvent fe) { }
public void focusLost (FocusEvent fe) {
if (txtBookId.getText().equals ("")) {
}
else {
id = Integer.parseInt (txtBookId.getText ());
long bookNo;
boolean found = false;
try {
ResultSet rs = st.executeQuery ("Select * from Books where BId="+id+"");   //Executing
the Query.
rs.next ();
bookNo = rs.getLong ("BId");
if (bookNo == id) {
found = true;
txtBookId.setText ("" + id);
txtBookName.setText ("" + rs.getString ("BName"));
mid=rs.getInt("Mid");
if(mid==0)
{
JOptionPane.showMessageDialog(this,"Not an Issued Book");
dispose();
}
else
{
ResultSet rs1=st.executeQuery("Select * from Members where id="+mid+"");
rs1.next();
txtIssued.setText ("" + rs1.getString (3));
bc=rs1.getInt("Bcnt");
due=rs1.getInt(6);
}
}
else {
found = false;
}
}
```

```
catch (SQLException sqlex) {
if (found == false) {
txtClear ();
JOptionPane.showMessageDialog (this, "Record not Found.");
}
}
}
}

private void txtClear () {
txtBookId.setText ("");
txtBookName.setText ("");
txtIssued.setText ("");
txtBookId.requestFocus ();
}
}
```

SearchBook.java

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

public class SearchBook extends JInternalFrame implements ActionListener {
JPanel pBook = new JPanel ();
JLabel lbSearch;
JRadioButton rb1,rb2,rb3,rb4;
JTextField txtSearch;
JButton btnFind, btnCancel;
int flag=0;
Statement st;
String bname,bauthor,bcat,search;
int bref,bmid,bid,rows=0;
JTable table;
JScrollPane jsp;
Object data1[][];
Container c;
public SearchBook (Connection con) {
```

```
super ("Search Books", false, true, false, true);
setSize (510, 300);
lbSearch = new JLabel ("Search Field");
lbSearch.setForeground (Color.black);
lbSearch.setBounds (15, 15, 100, 20);
txtSearch = new JTextField ();
txtSearch.setBounds (120, 15, 175, 25);
btnFind = new JButton ("Find Book");
btnFind.setBounds (25, 175, 125, 25);
btnFind.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (165, 175, 125, 25);
btnCancel.addActionListener (this);
rb1=new JRadioButton("By Title");
rb1.addActionListener(this);
rb1.setBounds (15, 45, 100, 20);
rb2=new JRadioButton("By Author");
rb2.addActionListener(this);
rb2.setBounds (15, 75, 100, 20);
rb3=new JRadioButton("By Category");
rb3.addActionListener(this);
rb3.setBounds (15, 105, 100, 20);
rb4=new JRadioButton("By id");
rb4.addActionListener(this);
rb4.setBounds(15,135,100,20);
pBook.setLayout (null);
pBook.add(lbSearch);
pBook.add(txtSearch);
pBook.add(btnFind);
pBook.add(btnCancel);
ButtonGroup bg=new ButtonGroup();
bg.add(rb1);
bg.add(rb2);
bg.add(rb3);
bg.add(rb4);
pBook.add(rb1);
pBook.add(rb2);
pBook.add(rb3);
pBook.add(rb4);
```

```
rb1.setSelected(true);
getContentPane().add (pBook, BorderLayout.CENTER);
c=getContentPane();
try
{
st = con.createStatement ();
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading Form.");
dispose ();
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == btnFind) {
if (txtSearch.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Search Field not Provided.");
txtSearch.requestFocus ();
}
else
{
String bname1,bauthor1,bcat1;
int num;
boolean found = false;
try {
String q,bavl,bisr;
num=st.executeUpdate("Delete * from BSearch");
ResultSet rs = st.executeQuery ("SELECT * FROM Books ");       //Executing the
Query.
search=txtSearch.getText();
search=search.toLowerCase();
while(rs.next())
{
bname=rs.getString(2);
bauthor=rs.getString("BAuthor");
bcat=rs.getString("BCat");
bref=rs.getInt("BRef");
if(bref==1) bisr="Yes";
```

```
else bisr="No";
bmid=rs.getInt("Mid");
if(bmid==0) bavl="Available";
else bavl="Issued:"+ bmid;
bid=rs.getInt("BId");

if(flag==0)
{
bname1=bname.toLowerCase();
if(bname1.equals(search)||(bname1.indexOf(search)!=-1))
{
System.out.println("Came Here2");
num=st.executeUpdate("insert into BSearch values("+bid+", '"+bname+"' , '"+bcat+"' ,
'"+bauthor+"' , '"+bavl+"', '"+bisr+"')");
rows++;
found=true;
}
}
else if(flag==1)
{
bauthor1=bauthor.toLowerCase();
if(bauthor1.equals(search)||(bauthor1.indexOf(search)!=-1))
{
num=st.executeUpdate("insert into BSearch values("+bid+", '"+bname+"' , '"+bcat+"' ,
'"+bauthor+"' , '"+bavl+"', '"+bisr+"')");
rows++;
found=true;
}
}
else if(flag==2)
{
bcat1=bcat.toLowerCase();
if(bcat1.equals(search)||(bcat1.indexOf(search)!=-1))
{
num=st.executeUpdate("insert into BSearch values("+bid+", '"+bname+"' , '"+bcat+"' ,
'"+bauthor+"' , '"+bavl+"', '"+bisr+"')");
rows++;
found=true;
}
```

```
}
else if(flag==3)
{
if(bid==Integer.parseInt(txtSearch.getText()))
{
rows++;
num=st.executeUpdate("insert into BSearch values("+bid+", '"+bname+"' , '"+bcat+"' ,
'"+bauthor+"' , '"+bavl+"', '"+bisr+"')");
found=true;
}
}
}
}
catch(SQLException sqlex) {
if (found == false) {
JOptionPane.showMessageDialog (this, "Record not Found.");
}
}
try{
data1=new Object[rows][6];
Object[] Colheads={"Book Id","Book
Name","Category","Author","Availability","Reference"};
ResultSet rs=st.executeQuery("Select * from BSearch");
for(int i1=0;i1<rows;i1++)
{
rs.next();
for(int j1=0;j1<6;j1++)
{
data1[i1][j1]=rs.getString(j1+1);
}
}
table=new JTable(data1,Colheads);
int v=ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED;
int h=ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED;
System.out.println("hai we came here");
jsp=new JScrollPane(table,v,h);
TableDisp td=new TableDisp(table);
```

```
}
catch(Exception sqlex) {
if (found == false) {
JOptionPane.showMessageDialog (this, "Some prob Found.");
}
}
}
}
if (obj == btnCancel) {
setVisible (false);
dispose();
}
if(obj==rb1)
{
flag=0;
}
if(obj==rb2)
{
flag=1;
}
if(obj==rb3)
{
flag=2;
}
if(obj==rb4)
{
flag=3;
}
}
}
```

SearchMember.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;
public class SearchMember extends JInternalFrame implements ActionListener
{
```

```
JPanel pBook = new JPanel ();
JLabel lbSearch;
JRadioButton rb1,rb2;
JTextField txtSearch;
JButton btnFind, btnCancel;
int flag=0,rows=0;
Statement st;
String mname,mcat,search;
JTable table;
Object data1[][];
Container c;
int mid,bcnt;
public SearchMember (Connection con)
{
super ("Search Members", false, true, false, true);
setSize (325, 250);
lbSearch = new JLabel ("Search Field");
lbSearch.setForeground (Color.black);
lbSearch.setBounds (15, 15, 100, 20);
txtSearch = new JTextField ();
txtSearch.setBounds (120, 15, 175, 25);
btnFind = new JButton ("Find Member");
btnFind.setBounds (25, 175, 125, 25);
btnFind.addActionListener (this);
btnCancel = new JButton ("Cancel");
btnCancel.setBounds (165, 175, 125, 25);
btnCancel.addActionListener (this);
rb1=new JRadioButton("By Id");
rb1.addActionListener(this);
rb1.setBounds (15, 45, 100, 20);
rb2=new JRadioButton("By Name");
rb2.addActionListener(this);
rb2.setBounds (15, 75, 100, 20);
pBook.setLayout (null);
pBook.add(lbSearch);
pBook.add(txtSearch);
pBook.add(btnFind);
pBook.add(btnCancel);
ButtonGroup bg=new ButtonGroup();
```

```
bg.add(rb1);
bg.add(rb2);
pBook.add(rb1);
pBook.add(rb2);
rb1.setSelected(true);
getContentPane().add (pBook, BorderLayout.CENTER);
try
{
st = con.createStatement ();
}
catch (SQLException sqlex) {
JOptionPane.showMessageDialog (null, "A Problem Occurs While Loading Form.");
dispose ();
}
setVisible (true);
}
public void actionPerformed (ActionEvent ae) {
Object obj = ae.getSource();
if (obj == btnFind) {
if (txtSearch.getText().equals ("")) {
JOptionPane.showMessageDialog (this, "Search Field not Provided.");
txtSearch.requestFocus ();
}
else
{
String mname1;
int num,id,catid,bcnt1;
boolean found = false;
ResultSet rs,rs1,rs3;
try {
String bavl,text,tts;
num=st.executeUpdate("Delete * from MSearch");
if(flag==0)
{
id=Integer.parseInt(txtSearch.getText());
rs=st.executeQuery("Select * from Members where id="+id+"");
rs.next();
bavl=rs.getString("Mname");
catid=rs.getInt(7);
```

```
bcnt=rs.getInt(5);
s1=st.executeQuery("Select * from MeCat where Mcat="+catid+"");
rs1.next();
mcat=rs1.getString("CName");
bcnt1=rs1.getInt("Blmt");
rs3=st.executeQuery("Select * from Books where Mid="+id+"");
text="Name: "+bavl+"\n Category: "+mcat+"\n Books Held: "+bcnt+"\n Book Limit:
"+bcnt1+"\n";
text+="Books Held:\n";
while(rs3.next())
{
tts=rs3.getString(2);
text+=tts+"\n";
}
JOptionPane.showMessageDialog(this,text);
txtSearch.setText("");
txtSearch.requestFocus();
}
else
{
search=txtSearch.getText();
search=search.toLowerCase();
rs=st.executeQuery("Select * from Members");
while(rs.next())
{
mname=rs.getString(3);
mid=rs.getInt(1);
bcnt=rs.getInt(5);
catid=rs.getInt(7);
if(flag==1)
{
mname1=mname.toLowerCase();
if(mname1.equals(search)||(mname1.indexOf(search)!=-1))
{
rs1=st.executeQuery("Select * from MeCat where Mcat="+catid+"");
rs1.next();
mcat=rs1.getString("CName");
bcnt1=rs1.getInt("Blmt");
num=st.executeUpdate("insert into MSearch values("+mid+", '"+mname+"' ,"+bcnt+",
```

```
'"+mcat+"',"+bcnt1+")");
rows++;
found=true;
}
}
}
}
catch(SQLException sqlex) {
if (found == false) {
JOptionPane.showMessageDialog (this, "Record not Found.");
}
}if(flag==1){
try{
data1=new Object[rows][5];
Object[] Colheads={"Member Id","Name","Books Held","Category","Book Limit"};
ResultSet rs2=st.executeQuery("Select * from MSearch");
for(int i1=0;i1<rows;i1++)
{
rs2.next();
for(int j1=0;j1<5;j1++)
{
data1[i1][j1]=rs2.getString(j1+1);
}
}
table=new JTable(data1,Colheads);
TableDisp td=new TableDisp(table);
txtSearch.setText("");
txtSearch.requestFocus();
}
catch(Exception sqlex) {
if (found == false) {
JOptionPane.showMessageDialog (this, "Some prob Found.");
}
}
}
}
}
if (obj == btnCancel) {
setVisible (false);
```

```
dispose();
}
if(obj==rb1)
{
flag=0;
}
if(obj==rb2)
{
flag=1;
}
}
}
```

TableDisp.java

```
import java.awt.*;
import javax.swing.*;
public class TableDisp extends JFrame
{
private JPanel pBook = new JPanel ();
private JScrollPane scroller;
private JTable table;
public TableDisp(JTable j)
{
super("Table Display");
setSize(500,300);
pBook.setLayout (null);
table=j;
scroller = new JScrollPane (table);
scroller.setBounds (20, 50, 460, 200);
pBook.add(scroller);
getContentPane().add (pBook, BorderLayout.CENTER);
setVisible(true);
}
}
```

public Last updated: 2019-07-03 05:10:28 AM

Comments

Your Name

Comment

Add Comment