

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



PRÁCTICA DE LABORATORIO

CARRERA: COMPUTACION

ASIGNATURA: PROGRAMACION APLICADA

NRO. PRÁCTICA:

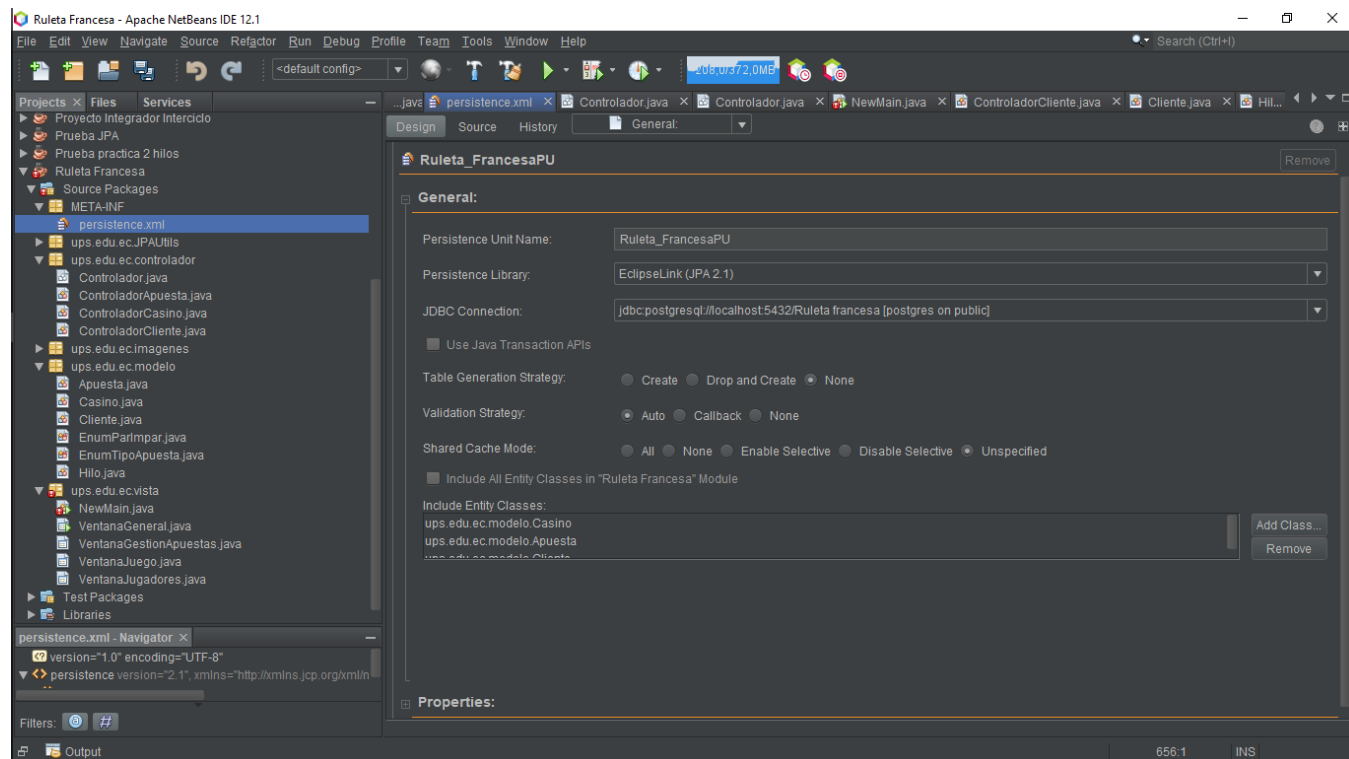
TÍTULO PRÁCTICA: EXAMEN FINAL

OBJETIVO ALCANZADO:

Consolidar los conocimientos adquiridos en clase sobre Java.

ACTIVIDADES DESARROLLADAS

1. JPA: 25%



	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

2. Hilos (Thread): 15 %

Para los hilos se creo una clase que implementa la interfaz Runnable la cual la llamé Hilo. A continuación, su código:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ups.edu.ec.modelo;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Collection;
import java.util.List;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.table.DefaultTableModel;
import ups.edu.ec.controlador.ControladorApuesta;
import ups.edu.ec.controlador.ControladorCasino;
import ups.edu.ec.controlador.ControladorCliente;
import ups.edu.ec.vista.VentanaJuego;

/**
 *
 * @author User
 */
public class Hilo implements Runnable {

    private Cliente cliente;

    private Casino casino;

    private int dineroApuesta;

    private int numeroPartida;

    private ControladorCliente controladorCliente;
    private ControladorApuesta controladorApuesta;
    private ControladorCasino controladorCasino;

    int contadorDineroApuesta = 0;

    private List<Apuesta> apuestas;
    private VentanaJuego ventanaJuego;

    public Hilo(Cliente cliente, Casino casino, int numeroPartida, ControladorCliente controladorCliente,
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

ControladorApuesta controladorApuesta, ControladorCasino controladorCasino, VentanaJuego
ventanaJuego) {
    this.cliente = cliente;
    this.casino = casino;
    this.dineroApuesta = 10;
    this.numeroPartida = numeroPartida;
    this.controladorCliente = controladorCliente;
    this.controladorApuesta = controladorApuesta;
    this.controladorCasino = controladorCasino;
    apuestas = new ArrayList<>();

    this.ventanaJuego = ventanaJuego;
}

public Cliente getCliente() {
    return cliente;
}

public void setCliente(Cliente cliente) {
    this.cliente = cliente;
}

public Casino getCasino() {
    return casino;
}

public void setCasino(Casino casino) {
    this.casino = casino;
}

public int getDineroApuesta() {
    return dineroApuesta;
}

public void setDineroApuesta(int dineroApuesta) {
    this.dineroApuesta = dineroApuesta;
}

public int getNumeroPartida() {
    return numeroPartida;
}

public void setNumeroPartida(int numeroPartida) {
    this.numeroPartida = numeroPartida;
}

public ControladorCliente getControladorCliente() {
    return controladorCliente;
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

}

public void setControladorCliente(ControladorCliente controladorCliente) {
    this.controladorCliente = controladorCliente;
}

public ControladorApuesta getControladorApuesta() {
    return controladorApuesta;
}

public void setControladorApuesta(ControladorApuesta controladorApuesta) {
    this.controladorApuesta = controladorApuesta;
}

public ControladorCasino getControladorCasino() {
    return controladorCasino;
}

public void setControladorCasino(ControladorCasino controladorCasino) {
    this.controladorCasino = controladorCasino;
}

public List<Apuesta> getApuestas() {
    return apuestas;
}

public void setApuestas(List<Apuesta> apuestas) {
    this.apuestas = apuestas;
}

public synchronized void llenarTblJuego() {
    DefaultTableModel modelo = (DefaultTableModel) ventanaJuego.getTblJuego().getModel();

    //modelo.setRowCount(0);
    for (int i = 0; i < apuestas.size() - 1; i++) {
        Apuesta ap = apuestas.get(i);
        Object[] row = {ap.getCodigoClienteFk().getNombre().concat(ap.getCodigoClienteFk().getApellido()),
            ap.getApostadoPara(), ap.getDineroCliente(), ap.getDineroCasino(), ap.getResultadoRuleta(),
            ap.getTipoApuesta()};
        modelo.addRow(row);
    }

    ventanaJuego.getTblJuego().setModel(modelo);
}

@Override
public void run() {
    if (bTieneDinero()) {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

Apuesta apuesta = new Apuesta();
switch (cliente.getTipoApuesta()) {

    case ("NUMEROCONCRETO") -> {
        //Apostamos siempre y cuando el hilo tenga dinero
        int iNumeroHilo = miApostarNumeroConcreto();
        //Dependiendo si coinciden los numeros o no, se hace una operacion u otra
        cliente.setNumeroApostado(String.valueOf(iNumeroHilo));
        if (Integer.valueOf(cliente.getNumeroRuleta()) == iNumeroHilo) {
            miNumeroGanado();
            System.out.println(Thread.currentThread().getName() + " ha ganado! Ahora tiene " +
cliente.getDinero() + " " + casino.getDinero());
            apuesta.setCantidadApuesta(10);
            apuesta.setCodigoClienteFk(cliente);
            apuesta.setGanador("Cliente");
            apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
            apuesta.setApostadoPara(String.valueOf(iNumeroHilo));
            apuesta.setTipoApuesta(cliente.getTipoApuesta());
            apuesta.setCodigo(getNumeroPartida());
            apuesta.setDineroCasino(casino.getDinero());
            apuesta.setDineroCliente(cliente.getDinero());

            System.out.println(apuesta);

        } else {

            apuesta.setCantidadApuesta(10);
            apuesta.setCodigoClienteFk(cliente);
            apuesta.setGanador("Casino");
            apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
            apuesta.setApostadoPara(String.valueOf(iNumeroHilo));
            apuesta.setTipoApuesta(cliente.getTipoApuesta());
            apuesta.setCodigo(getNumeroPartida());
            apuesta.setDineroCasino(casino.getDinero());
            apuesta.setDineroCliente(cliente.getDinero());
            System.out.println(Thread.currentThread().getName() + " ha perdido! Ahora tiene " +
cliente.getDinero() + " " + casino.getDinero());
            System.out.println(apuesta);
        }

        apuesta.setDineroCasino(casino.getDinero());
        apuesta.setDineroCliente(cliente.getDinero());
        apuesta.setCodigo(numeroPartida);
        /*controladorCliente.update(cliente);
        controladorCasino.update(casino);
        controladorApuesta.create(apuesta);*/

    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

case ("PARIMPAR") -> {
    String valor = (String.valueOf(miApostarParImpar()));
    cliente.setNumeroApostado(valor);
    if (Integer.valueOf(cliente.getNumeroRuleta()) % 2 == 0 && valor.equals("true")) {
        miNumeroGanado();
        System.out.println(Thread.currentThread().getName() + " ha ganado! Ahora tiene " +
cliente.getDinero() + " " + casino.getDinero());
        apuesta.setCantidadApuesta(10);
        apuesta.setCodigoClienteFk(cliente);
        apuesta.setGanador("Cliente");
        apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
        apuesta.setApostadoPara(String.valueOf(valor));
        apuesta.setTipoApuesta(cliente.getTipoApuesta());
        apuesta.setCodigo(getNumeroPartida());
        apuesta.setDineroCasino(casino.getDinero());
        apuesta.setDineroCliente(cliente.getDinero());
        System.out.println(apuesta);

    } else {
        //numeroPerdido();
        System.out.println(Thread.currentThread().getName() + " ha perdido! Ahora tiene " +
cliente.getDinero() + " " + casino.getDinero());
        apuesta.setCantidadApuesta(10);
        apuesta.setCodigoClienteFk(cliente);
        apuesta.setGanador("Casino");
        apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
        apuesta.setApostadoPara(String.valueOf(valor));
        apuesta.setTipoApuesta(cliente.getTipoApuesta());
        apuesta.setCodigo(getNumeroPartida());

        apuesta.setDineroCasino(casino.getDinero());
        apuesta.setDineroCliente(cliente.getDinero());

        System.out.println(apuesta);

    }
    apuesta.setDineroCasino(casino.getDinero());
    apuesta.setDineroCliente(cliente.getDinero());
    apuesta.setCodigo(numeroPartida);
    /*controladorCliente.update(cliente);
    controladorCasino.update(casino);
    controladorApuesta.create(apuesta);*/
}

default -> {
    int iNumeroHilo2 = miApostarMartinGala();
    cliente.setNumeroApostado(String.valueOf(iNumeroHilo2));
    System.out.println(getDineroApuesta());
    if (Integer.valueOf(cliente.getNumeroRuleta()) == iNumeroHilo2) {

```

```
miNumeroGanado();
System.out.println(Thread.currentThread().getName() + " ha ganado! Ahora tiene " +
cliente.getDinero() + " " + casino.getDinero());
apuesta.setCantidadApuesta(this.getDineroApuesta());
apuesta.setCodigoClienteFk(cliente);
apuesta.setGanador("Cliente");
apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
apuesta.setApostadoPara(String.valueOf(iNumeroHilo2));
apuesta.setTipoApuesta(cliente.getTipoApuesta());
apuesta.setCodigo(getNumeroPartida());
apuesta.setDineroCasino(casino.getDinero());
apuesta.setDineroCliente(cliente.getDinero());
System.out.println(apuesta);

} else {
    mvNumeroPerdidoMartinGala();
    System.out.println(Thread.currentThread().getName() + " ha perdido! Ahora tiene " +
cliente.getDinero() + " " + casino.getDinero());
apuesta.setCantidadApuesta(this.getDineroApuesta());
apuesta.setCodigoClienteFk(cliente);
apuesta.setGanador("Casino");
apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
apuesta.setApostadoPara(String.valueOf(iNumeroHilo2));
apuesta.setTipoApuesta(cliente.getTipoApuesta());
apuesta.setCodigo(getNumeroPartida());
apuesta.setDineroCasino(casino.getDinero());
apuesta.setDineroCliente(cliente.getDinero());
System.out.println(apuesta);
}
apuesta.setDineroCasino(casino.getDinero());
apuesta.setDineroCliente(cliente.getDinero());
apuesta.setCodigo(numeroPartida);

/*controladorCliente.update(cliente);
controladorCasino.update(casino);
controladorApuesta.create(apuesta);*/
}
}
anadirApuesta(apuesta);

try {
    Thread.sleep(500);

    //llenarTblJuego();
} catch (InterruptedException ex) {
    Logger.getLogger(Hilo.class.getName()).log(Level.SEVERE, null, ex);
}
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

}

public synchronized void anadirApuesta(Apuesta apue) {
    apuestas.add(apue);
}

public boolean bTieneDinero() {
    return cliente.getDinero() >= 10;
}

public int miApostarNumeroConcreto() {
    //Sacamos un numero al azar
    int iNumero = (int) (Math.random() * 36 + 1);
    //Quitamos el valor de la apuesta al hilo
    mvDisminuirDineroH(10);
    //Le damos el dinero al banco
    casino.setDinero(casino.getDinero() + 10);
    return iNumero;
}

private void miNumeroGanado() {
    //Le quitamos el dinero al banco dependiendo del hilo que sea
    //y del dinero que disponga el banco
    int iDineroDisponibleBanco = casino.getDineroGanado(Thread.currentThread());
    //Aumentamos el dinero al hilo
    mvAumentarDineroH(iDineroDisponibleBanco);
    //Le quitamos el dinero al banco
    casino.setDinero(casino.getDinero() - iDineroDisponibleBanco);
    //Aumentamos los beneficios del grupo
    //oBeneficios.setAumentoiEurosGrupo(iDineroDisponibleBanco);
}

public void mvAumentarDineroH(int iDinero) {
    int iNuevoDinero = cliente.getDinero() + iDinero;
    cliente.setDinero(iNuevoDinero);
}

public void mvDisminuirDineroH(int iDinero) {
    int iNuevoDinero = cliente.getDinero() - iDinero;
    cliente.setDinero(iNuevoDinero);
}

public String miApostarParImpar() {
    try {
        boolean bPar = Random.class.newInstance().nextBoolean();
        //Quitamos el valor de la apuesta al hilo
        mvDisminuirDineroH(10);
        //Le damos el dinero al banco
        casino.setDinero(casino.getDinero() + 10);
    }
}

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

//Devolvemos el valor buleano aleatorio
if (bPar) {
    return "true";
} else {
    return "false";
}

} catch (InstantiationException | IllegalAccessException ex) {
    Logger.getLogger(Cliente.class.getName()).log(Level.SEVERE, null, ex);
}

return "false";
}

public int miApostarMartinGala() {
    //Sacamos un numero al azar
    int iNumero = (int) (Math.random() * 36 + 1);
    //Quitamos el valor de la apuesta al hilo
    mvDisminuirDineroH(getDineroApuesta());
    //Le damos el dinero al banco
    casino.setDinero(casino.getDinero() + getDineroApuesta());

    return iNumero;
}

private void mvNumeroPerdidoMartinGala() {
    //oBeneficios.setAumentoiEurosGrupo(getiDineroApuesta());
    setDineroApuesta(getDineroApuesta() * 2);
}
}

```

A continuación, para que esta clase pueda utilizar los hilos, lo hicimos en la clase VentanaJuego cuando el usuario aplasta el botón para comenzar el juego. Se presenta el código.

```

private void btnComenzarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    comenzarBotones();

    Casino oBanco = new Casino();
    oBanco.setDinero(50000);
    oBanco.setNombre("Magno");

    int numeroRuleta = miSacarNumero();

    boolean bSeguir = true;

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

int iNumeroPartida = 0;

int contador = 0;

List<Hilo> listaHilosNumeroConceto = new ArrayList<>();
List<Hilo> listaHilosParImpar = new ArrayList<>();
List<Hilo> listaHilosMartinGala = new ArrayList<>();

for (int i = 0; i < 4; i++) {
    listaHilosNumeroConceto.add(new Hilo(listaNumeroConcreto.get(i),
        oBanco, iNumeroPartida, controladorCliente, controladorApuesta, controladorCasino, this));

    listaHilosParImpar.add(new Hilo(listaParImpar.get(i),
        oBanco, iNumeroPartida, controladorCliente, controladorApuesta, controladorCasino, this));

    listaHilosMartinGala.add(new Hilo(listaMartinGala.get(i),
        oBanco, iNumeroPartida, controladorCliente, controladorApuesta, controladorCasino, this));
}

while (bSeguir) {
    System.out.println(iNumeroPartida + "----- Numero Ruleta: " + iNumeroRuleta);

    if (iNumeroRuleta != 0 && contador != 2) {
        try {
            for (int i = 0; i < 4; i++) {
                System.out.println("hhhhhhhhhhhhhhhhhhhhhhhhhhhhhh");
                //Le pasamos el valor que ha sacado el crupier a los hilos
                listaNumeroConcreto.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));
                listaParImpar.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));
                listaMartinGala.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));

                listaHilosNumeroConceto.get(i).setNumeroPartida(iNumeroPartida);
                listaHilosParImpar.get(i).setNumeroPartida(iNumeroPartida);
                listaHilosMartinGala.get(i).setNumeroPartida(iNumeroPartida);
                listaHilos.add(listaHilosNumeroConceto.get(i));
                listaHilos.add(listaHilosParImpar.get(i));
                listaHilos.add(listaHilosMartinGala.get(i));

                //hilo3.setDineroApuesta(10);
                Thread th = new Thread(listaHilosNumeroConceto.get(i));
                Thread thPI = new Thread(listaHilosParImpar.get(i));
                Thread thMa = new Thread(listaHilosMartinGala.get(i));

                th.setName("Hilo " + i + " NCo");
                thPI.setName("Hilo " + i + " PI");
                thMa.setName("Hilo " + i + " MA");
                listaThreads.add(th);
                listaThreads.add(thPI);
                listaThreads.add(thMa);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParlImpar, listaHilosMartinGala);

//Inicializamos los hilos
th.start();
thPl.start();
thMa.start();
//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParlImpar, listaHilosMartinGala);
} // for()
iNumeroPartida++;
txtNumeroRuleta.setText(String.valueOf(iNumeroRuleta));
//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParlImpar, listaHilosMartinGala);
//Llamamos al metodo sleep para que se imprima la cuantia del banco justo despues de que se
ejecuten los hilos
Thread.sleep(50);
System.out.println("BANCO: " + oBanco.getDinero());

//Volvemos a utilizar el metodo sleep como se pide en el enunciado
Thread.sleep(3000);
//Sacamos otro numero pasados esos 3 segundos
iNumeroRuleta = miSacarNumero();
contador++;
} catch (InterruptedException ex) {
    Logger.getLogger(NewMain.class.getName()).log(Level.SEVERE, null, ex);
}
} else {
    bSeguir = false;
}
}

llenarTabla(listaHilos);
//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParlImpar, listaHilosMartinGala);

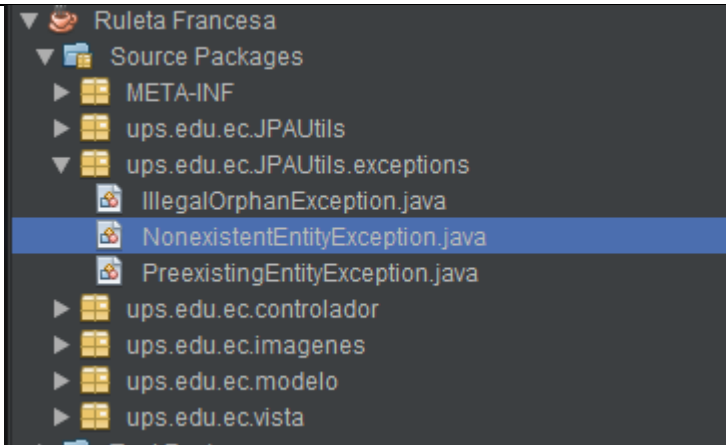
}

```

3. Excepciones: 5%

Las excepciones en este proyecto son principalmente por los controladores y por los hilos:

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



El código de las excepciones:

```
package ups.edu.ec.JPAUtils.exceptions;
```

```
import java.util.ArrayList;
import java.util.List;
```

```
public class IllegalOrphanException extends Exception {
    private List<String> messages;
    public IllegalOrphanException(List<String> messages) {
        super((messages != null && messages.size() > 0 ? messages.get(0) : null));
        if (messages == null) {
            this.messages = new ArrayList<String>();
        }
        else {
            this.messages = messages;
        }
    }
    public List<String> getMessages() {
        return messages;
    }
}
```

```
package ups.edu.ec.JPAUtils.exceptions;
```

```
public class NonexistentEntityException extends Exception {
    public NonexistentEntityException(String message, Throwable cause) {
        super(message, cause);
    }
    public NonexistentEntityException(String message) {
        super(message);
    }
}
```

```
package ups.edu.ec.JPAUtils.exceptions;
```

```
public class PreexistingEntityException extends Exception {  
    public PreexistingEntityException(String message, Throwable cause) {  
        super(message, cause);  
    }  
    public PreexistingEntityException(String message) {  
        super(message);  
    }  
}
```

Ahora el código donde se maneja los errores de los hilos:

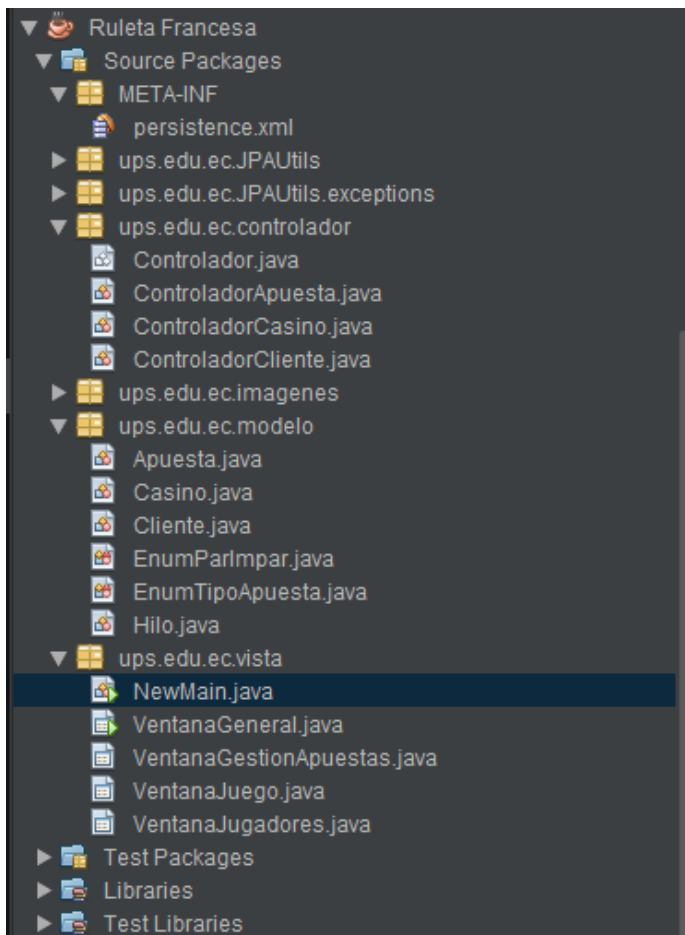
```
if (iNumeroRuleta != 0) {  
    try {  
        for (int i = 0; i < 4; i++) {  
            System.out.println("hhhhhhhhhhhhhhhhhhhhhhhhhhhhhh");  
            //Le pasamos el valor que ha sacado el crupier a los hilos  
            listaNumeroConcreto.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));  
            listaParImpar.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));  
            listaMartinGala.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));  
  
            listaHilosNumeroConcreto.get(i).setNumeroPartida(iNumeroPartida);  
            listaHilosParImpar.get(i).setNumeroPartida(iNumeroPartida);  
            listaHilosMartinGala.get(i).setNumeroPartida(iNumeroPartida);  
            listaHilos.add(listaHilosNumeroConcreto.get(i));  
            listaHilos.add(listaHilosParImpar.get(i));  
            listaHilos.add(listaHilosMartinGala.get(i));  
  
            //hilo3.setDineroApuesta(10);  
            Thread th = new Thread(listaHilosNumeroConcreto.get(i));  
            Thread thPI = new Thread(listaHilosParImpar.get(i));  
            Thread thMa = new Thread(listaHilosMartinGala.get(i));  
  
            th.setName("Hilo " + i + " NCo");  
            thPI.setName("Hilo " + i + " PI");  
            thMa.setName("Hilo " + i + " MA");  
            listaThreads.add(th);  
            listaThreads.add(thPI);  
            listaThreads.add(thMa);  
            //llenarTablaJuego(listaHilosNumeroConcreto, listaHilosParImpar, listaHilosMartinGala);  
  
            //Inicializamos los hilos  
            th.start();  
            thPI.start();  
            thMa.start();  
            //llenarTablaJuego(listaHilosNumeroConcreto, listaHilosParImpar, listaHilosMartinGala);  
        } // for()  
        iNumeroPartida++;  
        txtNumeroRuleta.setText(String.valueOf(iNumeroRuleta));  
    }  
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```
//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParImpar, listaHilosMartinGala);
//Llamamos al metodo sleep para que se imprima la cuantia del banco justo despues de que se
ejecuten los hilos
Thread.sleep(50);
System.out.println("BANCO: " + oBanco.getDinero());

//Volvemos a utilizar el metodo sleep como se pide en el enunciado
Thread.sleep(3000);
//Sacamos otro numero pasados esos 3 segundos
iNumeroRuleta = miSacarNumero();
contador++;
} catch (InterruptedException ex) {
    Logger.getLogger(NewMain.class.getName()).log(Level.SEVERE, null, ex);
}
```

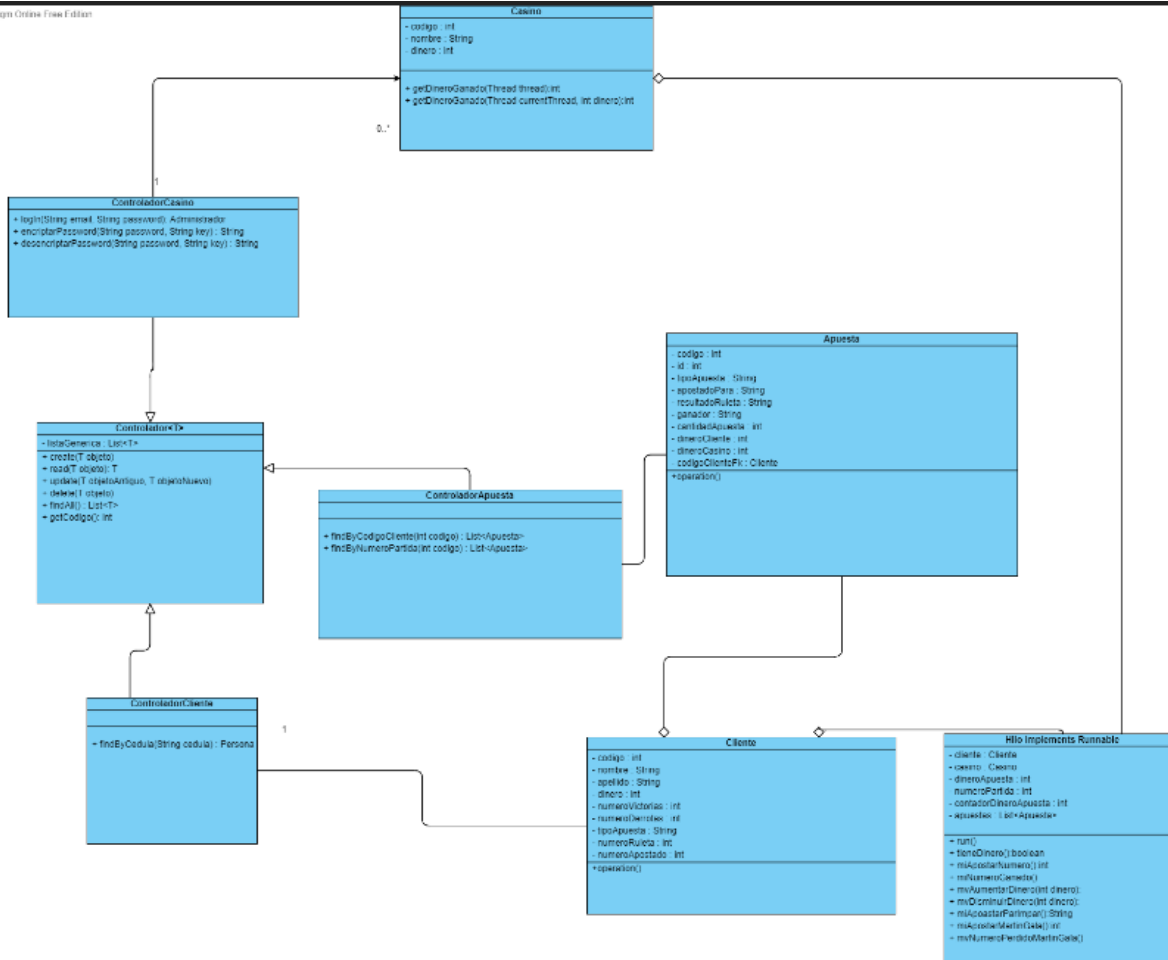
4. MVC: 10%



5. Diagrama de clases: 10%

Este es el diagrama UML del examen, se lo adjuntara al github para una mejor observación.

Visual Paradigm Online - New Edition

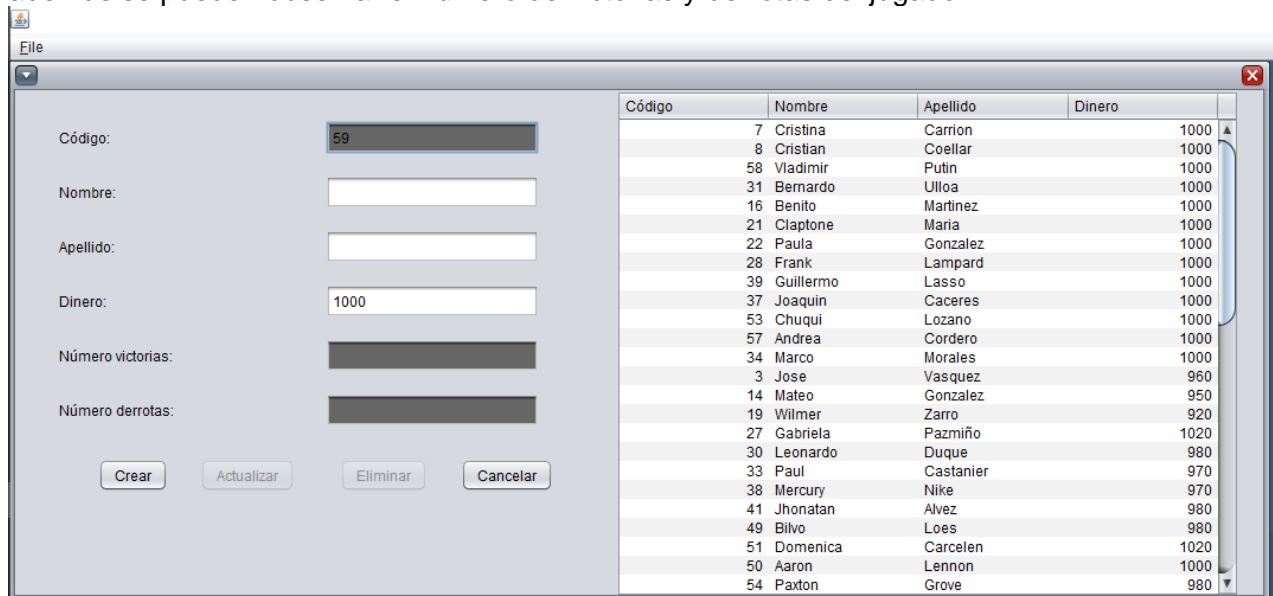


6. Usabilidad – Vista - Simulación: 25%

Cuando iniciamos el programa tenemos la siguiente ventana principal:



Para crear jugadores nos dirigimos a la ventana jugadores donde se pueden crear, editar, eliminar jugadores y además se pueden observar el número de victorias y derrotas del jugador



En este caso lo hicimos aplastando el botón de jugadores aleatorios:



Una observado los jugadores que se escogieron al azar, comienza el juego:

The software interface includes a roulette wheel, a control panel with buttons like 'Comenzar', 'Pau...', and 'Terminar', and a table displaying player information.


Código	Jugador	Juego
4	WalterMendoza	PARIMPAR
32	AlexMorgan	PARIMPAR
26	GaloCardenas	PARIMPAR
12	LolaGonzalez	MARTINGALA
10	CarolinaLandazuri	MARTINGALA
2	LorenaCarvalho	MARTINGALA
30	LeonardoDuque	MARTINGALA

Número Ruleta: 14

Jugador	Número	Dinero	Banco	Ruleta	Juego
GabrielaPazmi...	24	990	50060	14	NUMEROCON...
GabrielaPazmi...	22	990	50120	14	NUMEROCON...
BrexoWeverton	par	1010	50080	14	PARIMPAR
BrexoWeverton	impar	1000	50120	14	PARIMPAR
LolaGonzalez	13	990	50060	14	MARTINGALA
LolaGonzalez	29	970	50120	14	MARTINGALA
WilmerZarro	2	990	50060	14	NUMEROCON...
WilmerZarro	35	980	50130	14	NUMEROCON...
MateoHermida	impar	990	50070	14	PARIMPAR
MateoHermida	impar	980	50180	14	PARIMPAR
CarolinaLand...	34	990	50080	14	MARTINGALA
CarolinaLand...	28	970	50210	14	MARTINGALA
Warren Buffet	15	990	50060	14	NUMEROCON...
Warren Buffet	28	980	50140	14	NUMEROCON...
AlexMorgan	par	1010	50070	14	PARIMPAR
AlexMorgan	impar	1000	50200	14	PARIMPAR
LorenaCarvalho	17	990	50080	14	MARTINGALA
LorenaCarvalho	31	970	50210	14	MARTINGALA
PaulaGonzalez	9	990	50090	14	NUMEROCON...
PaulaGonzalez	17	980	50210	14	NUMEROCON...
GaloCardenas	impar	990	50060	14	PARIMPAR
GaloCardenas	par	1000	50200	14	PARIMPAR
LeonardoDuque	19	990	50060	14	MARTINGALA
LeonardoDuque	4	970	50220	14	MARTINGALA
GabrielaPazmi...	24	990	50060	14	NUMEROCON...
GabrielaPazmi...	22	980	50120	14	NUMEROCON...

Cuando deseamos que termine el juego aplastamos en el botón de terminar y el juego terminara.

```
if (!juegoPausado) {
    listaThreads.forEach(hi -> {
        hi.interrupt();
    });
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    } else {
        listaThreads.forEach(Thread::resume);
    }
}

private void btnTerminarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    try {
        listaThreads.forEach(hi -> {
            hi.stop();
        });

        System.out.println(listaJugadores);
        btnComenzar.setEnabled(false);
        btnParar.setEnabled(false);
        btnComenzar.setEnabled(false);
        btnJugadoresAleatorios.setEnabled(true);
        btnTerminar.setEnabled(false);
    } catch (Exception e) {
        e.printStackTrace();
    }
    //llenarTabla(listaHilos);
}

```

Ahora se muestra la ventana donde se pueden filtrar las apuestas ya sea por código del jugador o por numero de partida:

File

Busqueda: 49

CodigoCliente Buscar Cancelar

Código	Partida	Jugador	Numero	Ruleta	Dinero Apuesta	Dinero jugador	Dinero casino	Ganador	Tipo
62	1	BilvoLoes	par	34	10	1010	49340	Cliente	PARIMPAR
63	2	BilvoLoes	impar	35	10	1000	49090	Casino	PARIMPAR
86	84	BilvoLoes	par	34	10	1010	49240	Cliente	PARIMPAR
87	85	BilvoLoes	par	9	10	1000	49270	Casino	PARIMPAR
126	108	BilvoLoes	25	22	10	990	49480	Casino	NUMEROCO...
127	109	BilvoLoes	15	22	10	980	49550	Casino	NUMEROCO...

File

Busqueda: 2

NumeroApuesta Buscar Cancelar

Código	Partida	Jugador	Numero	Ruleta	Dinero Apuesta	Dinero jugador	Dinero casino	Ganador	Tipo
61	2	MariaLopez	22	35	10	980	49050	Casino	NUMEROCO...
63	2	BilvoLoes	impar	35	10	1000	49090	Casino	PARIMPAR
65	2	Ronaldo John...	23	35	40	970	49090	Casino	MARTINGALA
67	2	FernandoRoca	3	35	10	980	49080	Casino	NUMEROCO...
69	2	JoseVasquez	impar	35	10	1000	49120	Casino	PARIMPAR
71	2	MichaelEssien	10	35	40	970	49120	Casino	MARTINGALA
73	2	LionelMessi	13	35	10	980	49130	Casino	NUMEROCO...
75	2	Warren Buffet	impar	35	10	980	49140	Casino	PARIMPAR
77	2	WilmerZarro	3	35	40	970	49180	Casino	MARTINGALA
79	2	JuanUrdiales	5	35	10	980	49190	Casino	NUMEROCO...
81	2	MateoGonzalez	par	35	10	980	49180	Casino	PARIMPAR
83	2	ComisarioCat...	28	35	20	1340	49190	Casino	MARTINGALA

7. Programación genérica, Java 8, reflexión: 10%

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Controlador abstracto:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

```
package ups.edu.ec.controlador;
```

```
import java.lang.reflect.ParameterizedType;
import java.util.List;
import javax.persistence.EntityManager;
import ups.edu.ec.JPAUtils.JPAUtils;
```

```
/**
 *
 * @author User
 * @param <E>
 */
```

```
public abstract class Controlador<E> {
```

```
    private EntityManager em;
    private Class<E> clase;
```

```
    public Controlador() {
        this.em = (EntityManager) JPAUtils.getEmf();

        java.lang.reflect.Type t = getClass().getGenericSuperclass();
        ParameterizedType pt = (ParameterizedType) t;

        clase = (Class) pt.getActualTypeArguments()[0];
    }
```

```
    public EntityManager getEm() {
        return em;
    }
```

```
    public void setEm(EntityManager em) {
        this.em = em;
    }
```

```
    public boolean create(E objeto) {
```

```
        try {
```

```
            em.getTransaction().begin();
            em.persist(objeto);
            em.getTransaction().commit();
```

```
        return true;
    } catch (Exception e) {
        e.printStackTrace();
        return false;
    }
}

public E read(int id) {
    return em.find(clase, id);
}

public boolean update(E objeto) {
    em.getTransaction().begin();
    em.merge(objeto);
    em.getTransaction().commit();
    return true;
}

public boolean delete(E objeto) {
    em.getTransaction().begin();
    if (!em.contains(objeto)) {
        objeto = em.merge(objeto);
    }

    em.remove(objeto);
    em.getTransaction().commit();
    return true;
}

public abstract List<E> findAll();

public abstract int getCodigo();
}
```

Java 8:

```

hilos.forEach(cl -> {
    cl.getApuestas().stream().map(ap -> {
        String numeroApostado = "";
        if (ap.getApostadoPara().equalsIgnoreCase("true")) {
            numeroApostado = "par";
        } else if ((ap.getApostadoPara().equalsIgnoreCase("false"))) {
            numeroApostado = "impar";
        } else {
            numeroApostado = ap.getApostadoPara();
        }
        ap.setApostadoPara(numeroApostado);
        Object[] row = {ap.getCodigoClienteFk().getNombre().concat(ap.getCodigoClienteFk().getApellido()),
            numeroApostado, ap.getDineroCliente(), ap.getDineroCasino(), ap.getResultadoRuleta(),
            ap.getTipoApuesta()};
        return row;
    }).forEachOrdered(row -> {
        modelo.addRow(row);
    });
});

```

```

public int numeroPartida() {
    var lista = findAll();

    if (lista.isEmpty()) {
        return 0;
    } else {
        Collections.sort(lista, (Apuesta ap1, Apuesta ap2) -> ap1.getId().compareTo(ap2.getId()));
        return lista.get(lista.size() - 1).getCodigo();
    }
}

```

Reflexión:

```

public Controlador() {
    this.em = (EntityManager) JPAUtils.getEmf();

    java.lang.reflect.Type t = getClass().getGenericSuperclass();
    ParameterizedType pt = (ParameterizedType) t;

    clase = (Class) pt.getActualTypeArguments()[0];
}

```

2. Código del proyecto

Paquete modelo:

Clase apuesta:

/*

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

* and open the template in the editor.

*/

package ups.edu.ec.modelo;

import java.io.Serializable;

import javax.persistence.Basic;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

import javax.persistence.NamedQueries;

import javax.persistence.NamedQuery;

import javax.persistence.Table;

/**

*

* @author User

*/

@Entity

@Table(name = "apuesta")

@NamedQueries({

 @NamedQuery(name = "Apuesta.findAll", query = "SELECT a FROM Apuesta a"),

 @NamedQuery(name = "Apuesta.findById", query = "SELECT a FROM Apuesta a WHERE a.id = :id"),

 @NamedQuery(name = "Apuesta.findByCodigo", query = "SELECT a FROM Apuesta a WHERE a.codigo = :codigo"),

 @NamedQuery(name = "Apuesta.findByTipoApuesta", query = "SELECT a FROM Apuesta a WHERE a.tipoApuesta = :tipoApuesta"),

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

@NamedQuery(name = "Apuesta.findByApostadoPara", query = "SELECT a FROM Apuesta a WHERE a.apostadoPara = :apostadoPara"),

@NamedQuery(name = "Apuesta.findByResultadoRuleta", query = "SELECT a FROM Apuesta a WHERE a.resultadoRuleta = :resultadoRuleta"),

@NamedQuery(name = "Apuesta.findByGanador", query = "SELECT a FROM Apuesta a WHERE a.ganador = :ganador"),

@NamedQuery(name = "Apuesta.findByCantidadApuesta", query = "SELECT a FROM Apuesta a WHERE a.cantidadApuesta = :cantidadApuesta"),

@NamedQuery(name = "Apuesta.findByDineroCliente", query = "SELECT a FROM Apuesta a WHERE a.dineroCliente = :dineroCliente"),

@NamedQuery(name = "Apuesta.findByDineroCasino", query = "SELECT a FROM Apuesta a WHERE a.dineroCasino = :dineroCasino"))

public class Apuesta implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    @Basic(optional = false)

    @Column(name = "id")

    private Integer id;

    @Column(name = "codigo")

    private Integer codigo;

    @Column(name = "tipo_apuesta")

    private String tipoApuesta;

    @Column(name = "apostado_para")

    private String apostadoPara;

    @Column(name = "resultado_ruleta")

    private String resultadoRuleta;

    @Column(name = "ganador")

    private String ganador;

    @Column(name = "cantidad_apuesta")

    private Integer cantidadApuesta;

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

@Column(name = "dinero_cliente")

private Integer dineroCliente;

@Column(name = "dinero_casino")

private Integer dineroCasino;

@JoinColumn(name = "codigo_cliente_fk", referencedColumnName = "codigo")

@ManyToOne

private Cliente codigoClienteFk;


public Apuesta() {
}

public Apuesta(Integer id) {
    this.id = id;
}


public Integer getId() {
    return id;
}

public void setId(Integer id) {
    this.id = id;
}

public Integer getCodigo() {
    return codigo;
}

public void setCodigo(Integer codigo) {
    this.codigo = codigo;
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

}

public String getTipoApuesta() {
    return tipoApuesta;
}

public void setTipoApuesta(String tipoApuesta) {
    this.tipoApuesta = tipoApuesta;
}

public String getApostadoPara() {
    return apostadoPara;
}


public void setApostadoPara(String apostadoPara) {
    this.apostadoPara = apostadoPara;
}

public String getResultadoRuleta() {
    return resultadoRuleta;
}

public void setResultadoRuleta(String resultadoRuleta) {
    this.resultadoRuleta = resultadoRuleta;
}

public String getGanador() {
    return ganador;
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public void setGanador(String ganador) {
    this.ganador = ganador;
}

public Integer getCantidadApuesta() {
    return cantidadApuesta;
}

public void setCantidadApuesta(Integer cantidadApuesta) {
    this.cantidadApuesta = cantidadApuesta;
}

public Integer getDineroCliente() {
    return dineroCliente;
}

public void setDineroCliente(Integer dineroCliente) {
    this.dineroCliente = dineroCliente;
}

public Integer getDineroCasino() {
    return dineroCasino;
}

public void setDineroCasino(Integer dineroCasino) {
    this.dineroCasino = dineroCasino;
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public Cliente getCodigoClienteFk() {
    return codigoClienteFk;
}

public void setCodigoClienteFk(Cliente codigoClienteFk) {
    this.codigoClienteFk = codigoClienteFk;
}

@Override
public int hashCode() {
    int hash = 0;
    hash += (id != null ? id.hashCode() : 0);
    return hash;
}

@Override
public boolean equals(Object object) {
    // TODO: Warning - this method won't work in the case the id fields are not set
    if (!(object instanceof Apuesta)) {
        return false;
    }
    Apuesta other = (Apuesta) object;
    if ((this.id == null && other.id != null) || (this.id != null && !this.id.equals(other.id))) {
        return false;
    }
    return true;
}

@Override

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public String toString() {

    return "Apuesta{" + "id=" + id + ", codigo=" + codigo + ", tipoApuesta=" + tipoApuesta + ", apostadoPara=" +
apostadoPara + ", resultadoRuleta=" + resultadoRuleta + ", ganador=" + ganador + ", cantidadApuesta=" + cantidadApuesta +
", dineroCliente=" + dineroCliente + ", dineroCasino=" + dineroCasino + ", codigoClienteFk=" + codigoClienteFk + '}';

}

```

```

}

```

Clase Casino:

```


/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package ups.edu.ec.modelo;

import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;

/**

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

*
* @author User
*/
@Entity
@Table(name = "casino")
@NamedQueries({
    @NamedQuery(name = "Casino.findAll", query = "SELECT c FROM Casino c"),
    @NamedQuery(name = "Casino.findByCodigo", query = "SELECT c FROM Casino c WHERE c.codigo = :codigo"),
    @NamedQuery(name = "Casino.findByNombre", query = "SELECT c FROM Casino c WHERE c.nombre = :nombre"),
    @NamedQuery(name = "Casino.findByDinero", query = "SELECT c FROM Casino c WHERE c.dinero = :dinero"))
public class Casino implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "codigo")
    private Integer codigo;


    @Column(name = "nombre")
    private String nombre;

    @Column(name = "dinero")
    private Integer dinero;

    public Casino() {
    }

    public Casino(Integer codigo) {
        this.codigo = codigo;
    }

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public Integer getCodigo() {
    return codigo;
}

public void setCodigo(Integer codigo) {
    this.codigo = codigo;
}

public String getNombre() {
    return nombre;
}


public void setNombre(String nombre) {
    this.nombre = nombre;
}

public synchronized Integer getDinero() {
    return dinero;
}

public synchronized void setDinero(Integer dinero) {
    this.dinero = dinero;
}

public synchronized int getDineroGanado(Thread thread) {
    if (thread.getName().contains("NCo")) //Es un hilo de estrategia numero concreto
    {
        if (getDinero() > 360) {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```


        return 360;
    } else {
        return 0;
    }
} else if (thread.getName().contains("PI")) //Hilo par impar
{
    if (getDinero() > 20) {
        return 20;
    } else {
        return 0;
    }
} else {
    return 360;
}

}

public int getDineroGanado(Thread currentThread, int iiDineroApuesta) {
    //Hilo Maringala
    if (getDinero() > iiDineroApuesta * 36) {
        return iiDineroApuesta * 36;
    } else {
        return 0;
    }
}

@Override
public int hashCode() {
    int hash = 0;

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

hash += (codigo != null ? codigo.hashCode() : 0);

return hash;

}

@Override

public boolean equals(Object object) {

    // TODO: Warning - this method won't work in the case the id fields are not set

    if (!(object instanceof Casino)) {

        return false;

    }

    Casino other = (Casino) object;

    if ((this.codigo == null && other.codigo != null) || (this.codigo != null && !this.codigo.equals(other.codigo))) {

        return false;

    }

    return true;

}

@Override

public String toString() {

    return "ups.edu.ec.modelo.Casino[ codigo=" + codigo + " ]";

}

}

```

Clase Cliente:

/*

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

* and open the template in the editor.

*/

package ups.edu.ec.modelo;

import java.io.Serializable;

import java.util.Collection;

import javax.persistence.Basic;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.NamedQueries;

import javax.persistence.NamedQuery;

import javax.persistence.OneToOne;

import javax.persistence.Table;

/**

*

* @author User

*/

@Entity

@Table(name = "cliente")

@NamedQueries({

 @NamedQuery(name = "Cliente.findAll", query = "SELECT c FROM Cliente c"),

 @NamedQuery(name = "Cliente.findByCodigo", query = "SELECT c FROM Cliente c WHERE c.codigo = :codigo"),

 @NamedQuery(name = "Cliente.findByNombre", query = "SELECT c FROM Cliente c WHERE c.nombre = :nombre"),

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

@NamedQuery(name = "Cliente.findByApellido", query = "SELECT c FROM Cliente c WHERE c.apellido = :apellido"),
@NamedQuery(name = "Cliente.findByDinero", query = "SELECT c FROM Cliente c WHERE c.dinero = :dinero"),
@NamedQuery(name = "Cliente.findByNumeroVictorias", query = "SELECT c FROM Cliente c WHERE c.numeroVictorias = :numeroVictorias"),
@NamedQuery(name = "Cliente.findByNumeroDerrotas", query = "SELECT c FROM Cliente c WHERE c.numeroDerrotas = :numeroDerrotas"),
@NamedQuery(name = "Cliente.findByTipoApuesta", query = "SELECT c FROM Cliente c WHERE c.tipoApuesta = :tipoApuesta"),
@NamedQuery(name = "Cliente.findByNumeroRuleta", query = "SELECT c FROM Cliente c WHERE c.numeroRuleta = :numeroRuleta"))

public class Cliente implements Serializable {

    @Column(name = "numero_apostado")
    private String numeroApostado;

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "codigo")
    private Integer codigo;

    @Column(name = "nombre")
    private String nombre;

    @Column(name = "apellido")
    private String apellido;

    @Column(name = "dinero")
    private Integer dinero;

    @Column(name = "numero_victorias")
    private Integer numeroVictorias;

    @Column(name = "numero_derrotas")

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

private Integer numeroDerrotas;

@Column(name = "tipo_apuesta")
private String tipoApuesta;

@Column(name = "numero_ruleta")
private String numeroRuleta;

@OneToMany(mappedBy = "codigoClienteFk")
private Collection<Apuesta> apuestaCollection;

public Cliente() {
}

public Cliente(String nombre, String apellido, String tipoApuesta) {
    this.nombre = nombre;
    this.apellido = apellido;
    this.tipoApuesta = tipoApuesta;
    this.dinero = 1000;
    this.numeroDerrotas = 0;
    this.numeroVictorias = 0;
}

public Cliente(String nombre, String apellido) {
    this.nombre = nombre;
    this.apellido = apellido;
    this.dinero = 1000;
    this.numeroDerrotas = 0;
    this.numeroVictorias = 0;
    this.numeroRuleta = "-1";
    this.tipoApuesta = "";
    this.numeroApostado = "-1";
}

```

```
}

public Cliente(Integer codigo) {
    this.codigo = codigo;
}

public Integer getCodigo() {
    return codigo;
}


public void setCodigo(Integer codigo) {
    this.codigo = codigo;
}

public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public String getApellido() {
    return apellido;
}

public void setApellido(String apellido) {
    this.apellido = apellido;
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public Integer getDinero() {
    return dinero;
}

public void setDinero(Integer dinero) {
    this.dinero = dinero;
}

public Integer getNumeroVictorias() {
    return numeroVictorias;
}


public void setNumeroVictorias(Integer numeroVictorias) {
    this.numeroVictorias = numeroVictorias;
}

public Integer getNumeroDerrotas() {
    return numeroDerrotas;
}

public void setNumeroDerrotas(Integer numeroDerrotas) {
    this.numeroDerrotas = numeroDerrotas;
}

public String getTipoApuesta() {
    return tipoApuesta;
}

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public void setTipoApuesta(String tipoApuesta) {
    this.tipoApuesta = tipoApuesta;
}

public String getNumeroRuleta() {
    return numeroRuleta;
}

public void setNumeroRuleta(String numeroRuleta) {
    this.numeroRuleta = numeroRuleta;
}


public Collection<Apuesta> getApuestaCollection() {
    return apuestaCollection;
}

public void setApuestaCollection(Collection<Apuesta> apuestaCollection) {
    this.apuestaCollection = apuestaCollection;
}

@Override
public int hashCode() {
    int hash = 0;
    hash += (codigo != null ? codigo.hashCode() : 0);
    return hash;
}

@Override
public boolean equals(Object object) {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```
// TODO: Warning - this method won't work in the case the id fields are not set
if (!(object instanceof Cliente)) {
    return false;
}
Cliente other = (Cliente) object;
if ((this.codigo == null && other.codigo != null) || (this.codigo != null && !this.codigo.equals(other.codigo))) {
    return false;
}
return true;
}

@Override
public String toString() {
    return "Cliente{" + "numeroApostado=" + numeroApostado + ", codigo=" + codigo + ", nombre=" + nombre + ", apellido="
+ apellido + ", dinero=" + dinero + ", numeroVictorias=" + numeroVictorias + ", numeroDerrotas=" + numeroDerrotas + ",
tipoApuesta=" + tipoApuesta + ", numeroRuleta=" + numeroRuleta + ", apuestaCollection=" + apuestaCollection + '}';
}

public String getNumeroApostado() {
    return numeroApostado;
}

public void setNumeroApostado(String numeroApostado) {
    this.numeroApostado = numeroApostado;
}
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Clase Hilo:

```


/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ups.edu.ec.modelo;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Collection;
import java.util.List;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.table.DefaultTableModel;
import ups.edu.ec.controlador.ControladorApuesta;
import ups.edu.ec.controlador.ControladorCasino;
import ups.edu.ec.controlador.ControladorCliente;
import ups.edu.ec.vista.VentanaJuego;

/**
 *
 * @author User
 */
public class Hilo implements Runnable {

    private Cliente cliente;

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

private Casino casino;

private int dineroApuesta;

private int numeroPartida;

private ControladorCliente controladorCliente;

private ControladorApuesta controladorApuesta;

private ControladorCasino controladorCasino;

int contadorDineroApuesta = 0;

private List<Apuesta> apuestas;

private VentanaJuego ventanaJuego;

```
public Hilo(Cliente cliente, Casino casino, int numeroPartida, ControladorCliente controladorCliente,
    ControladorApuesta controladorApuesta, ControladorCasino controladorCasino, VentanaJuego ventanaJuego) {
    this.cliente = cliente;
    this.casino = casino;
    this.dineroApuesta = 10;
    this.numeroPartida = numeroPartida;
    this.controladorCliente = controladorCliente;
    this.controladorApuesta = controladorApuesta;
    this.controladorCasino = controladorCasino;
    apuestas = new ArrayList<>();

    this.ventanaJuego = ventanaJuego;
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

}

public Cliente getCliente() {
    return cliente;
}

public void setCliente(Cliente cliente) {
    this.cliente = cliente;
}

public Casino getCasino() {
    return casino;
}


public void setCasino(Casino casino) {
    this.casino = casino;
}

public int getDineroApuesta() {
    return dineroApuesta;
}

public void setDineroApuesta(int dineroApuesta) {
    this.dineroApuesta = dineroApuesta;
}

public int getNumeroPartida() {
    return numeroPartida;
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public void setNumeroPartida(int numeroPartida) {
    this.numeroPartida = numeroPartida;
}

public ControladorCliente getControladorCliente() {
    return controladorCliente;
}

public void setControladorCliente(ControladorCliente controladorCliente) {
    this.controladorCliente = controladorCliente;
}

public ControladorApuesta getControladorApuesta() {
    return controladorApuesta;
}

public void setControladorApuesta(ControladorApuesta controladorApuesta) {
    this.controladorApuesta = controladorApuesta;
}

public ControladorCasino getControladorCasino() {
    return controladorCasino;
}

public void setControladorCasino(ControladorCasino controladorCasino) {
    this.controladorCasino = controladorCasino;
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public List<Apuesta> getApuestas() {
    return apuestas;
}

public void setApuestas(List<Apuesta> apuestas) {
    this.apuestas = apuestas;
}

public synchronized void llenarTblJuego() {
    DefaultTableModel modelo = (DefaultTableModel) ventanaJuego.getTblJuego().getModel();

    //modelo.setRowCount(0);
    for (int i = 0; i < apuestas.size() - 1; i++) {
        Apuesta ap = apuestas.get(i);
        Object[] row = {ap.getCodigoClienteFk().getNombre().concat(ap.getCodigoClienteFk().getApellido()),
            ap.getApostadoPara(), ap.getDineroCliente(), ap.getDineroCasino(), ap.getResultadoRuleta(),
            ap.getTipoApuesta()};
        modelo.addRow(row);
    }

    ventanaJuego.getTblJuego().setModel(modelo);
}

@Override
public void run() {
    if (bTieneDinero()) {

        Apuesta apuesta = new Apuesta();
        switch (cliente.getTipoApuesta()) {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

case ("NUMEROCONCRETO") -> {

    //Apostamos siempre y cuando el hilo tenga dinero
    int iNumeroHilo = miApostarNumeroConcreto();

    //Dependiendo si coinciden los numeros o no, se hace una operacion u otra
    cliente.setNumeroApostado(String.valueOf(iNumeroHilo));

    if (Integer.valueOf(cliente.getNumeroRuleta()) == iNumeroHilo) {

        miNumeroGanado();

        System.out.println(Thread.currentThread().getName() + " ha ganado! Ahora tiene " + cliente.getDinero() + " " +
casino.getDinero());

        apuesta.setCantidadApuesta(10);
        apuesta.setCodigoClienteFk(cliente);
        apuesta.setGanador("Cliente");
        apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
        apuesta.setApostadoPara(String.valueOf(iNumeroHilo));
        apuesta.setTipoApuesta(cliente.getTipoApuesta());
        apuesta.setCodigo(getNumeroPartida());
        apuesta.setDineroCasino(casino.getDinero());
        apuesta.setDineroCliente(cliente.getDinero());

        System.out.println(apuesta);

    } else {

        cliente.setNumeroDerrotas(cliente.getNumeroDerrotas() + 1);
        apuesta.setCantidadApuesta(10);
        apuesta.setCodigoClienteFk(cliente);
        apuesta.setGanador("Casino");
        apuesta.setResultadoRuleta(cliente.getNumeroRuleta());

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        apuesta.setApostadoPara(String.valueOf(iNumeroHilo));
        apuesta.setTipoApuesta(cliente.getTipoApuesta());
        apuesta.setCodigo(getNumeroPartida());
        apuesta.setDineroCasino(casino.getDinero());
        apuesta.setDineroCliente(cliente.getDinero());

        System.out.println(Thread.currentThread().getName() + " ha perdido! Ahora tiene " + cliente.getDinero() + " " +
casino.getDinero());

        System.out.println(apuesta);

    }

    apuesta.setDineroCasino(casino.getDinero());
    apuesta.setDineroCliente(cliente.getDinero());
    apuesta.setCodigo(numeroPartida);
    /*controladorCliente.update(cliente);
    controladorCasino.update(casino);
    controladorApuesta.create(apuesta);*/

}

case ("PARIMPAR") -> {
    String valor = (String.valueOf(miApostarParImpar()));
    cliente.setNumeroApostado(valor);
    if (Integer.valueOf(cliente.getNumeroRuleta()) % 2 == 0 && valor.equals("true")) {
        miNumeroGanado();

        System.out.println(Thread.currentThread().getName() + " ha ganado! Ahora tiene " + cliente.getDinero() + " " +
casino.getDinero());

        apuesta.setCantidadApuesta(10);
        apuesta.setCodigoClienteFk(cliente);
        apuesta.setGanador("Cliente");
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
apuesta.setApostadoPara(String.valueOf(valor));
apuesta.setTipoApuesta(cliente.getTipoApuesta());
apuesta.setCodigo(getNumeroPartida());
apuesta.setDineroCasino(casino.getDinero());
apuesta.setDineroCliente(cliente.getDinero());
System.out.println(apuesta);

} else {
    //numeroPerdido();
    cliente.setNumeroDerrotas(cliente.getNumeroDerrotas() + 1);
    System.out.println(Thread.currentThread().getName() + " ha perdido! Ahora tiene " + cliente.getDinero() + " " +
casino.getDinero());

    apuesta.setCantidadApuesta(10);
    apuesta.setCodigoClienteFk(cliente);
    apuesta.setGanador("Casino");
    apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
    apuesta.setApostadoPara(String.valueOf(valor));
    apuesta.setTipoApuesta(cliente.getTipoApuesta());
    apuesta.setCodigo(getNumeroPartida());

    apuesta.setDineroCasino(casino.getDinero());
    apuesta.setDineroCliente(cliente.getDinero());

    System.out.println(apuesta);

}

apuesta.setDineroCasino(casino.getDinero());
apuesta.setDineroCliente(cliente.getDinero());

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

apuesta.setCodigo(numeroPartida);
/*controladorCliente.update(cliente);
controladorCasino.update(casino);
controladorApuesta.create(apuesta);*/
}

```

```

default -> {

    int iNumeroHilo2 = miApostarMartinGala();
    cliente.setNumeroApostado(String.valueOf(iNumeroHilo2));
    System.out.println(getDineroApuesta());
    if (Integer.valueOf(cliente.getNumeroRuleta()) == iNumeroHilo2) {
        miNumeroGanado();

        System.out.println(Thread.currentThread().getName() + " ha ganado! Ahora tiene " + cliente.getDinero() + " " +
casino.getDinero());

        apuesta.setCantidadApuesta(this.getDineroApuesta());
        apuesta.setCodigoClienteFk(cliente);
        apuesta.setGanador("Cliente");
        apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
        apuesta.setApostadoPara(String.valueOf(iNumeroHilo2));
        apuesta.setTipoApuesta(cliente.getTipoApuesta());
        apuesta.setCodigo(getNumeroPartida());
        apuesta.setDineroCasino(casino.getDinero());
        apuesta.setDineroCliente(cliente.getDinero());
        System.out.println(apuesta);

    } else {

        mvNumeroPerdidoMartinGala();
        cliente.setNumeroDerrotas(cliente.getNumeroDerrotas() + 1);

        System.out.println(Thread.currentThread().getName() + " ha perdido! Ahora tiene " + cliente.getDinero() + " " +
casino.getDinero());
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

apuesta.setCantidadApuesta(this.getDineroApuesta());
apuesta.setCodigoClienteFk(cliente);
apuesta.setGanador("Casino");
apuesta.setResultadoRuleta(cliente.getNumeroRuleta());
apuesta.setApostadoPara(String.valueOf(iNumeroHilo2));
apuesta.setTipoApuesta(cliente.getTipoApuesta());
apuesta.setCodigo(getNumeroPartida());
apuesta.setDineroCasino(casino.getDinero());
apuesta.setDineroCliente(cliente.getDinero());
System.out.println(apuesta);

}

apuesta.setDineroCasino(casino.getDinero());
apuesta.setDineroCliente(cliente.getDinero());
apuesta.setCodigo(numeroPartida);


/*controladorCliente.update(cliente);
controladorCasino.update(casino);
controladorApuesta.create(apuesta);*/
}
}

anadirApuesta(apuesta);

try {
    Thread.sleep(500);

    //llenarTblJuego();
} catch (InterruptedException ex) {
    Logger.getLogger(Hilo.class.getName()).log(Level.SEVERE, null, ex);

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    }

    }

    }

    public synchronized void anadirApuesta(Apuesta apue) {
        apuestas.add(apue);
    }

    public boolean bTieneDinero() {
        return cliente.getDinero() >= 10;
    }

    public int miApostarNumeroConcreto() {
        //Sacamos un numero al azar
        int iNumero = (int) (Math.random() * 36 + 1);
        //Quitamos el valor de la apuesta al hilo
        mvDisminuirDineroH(10);
        //Le damos el dinero al banco
        casino.setDinero(casino.getDinero() + 10);
        return iNumero;
    }

    private void miNumeroGanado() {
        //Le quitamos el dinero al banco dependiendo del hilo que sea
        //y del dinero que disponga el banco
        int iDineroDisponibleBanco = casino.getDineroGanado(Thread.currentThread());
        //Aumentamos el dinero al hilo
        mvAumentarDineroH(iDineroDisponibleBanco);
    }

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		


```
//Le quitamos el dinero al banco
casino.setDinero(casino.getDinero() - iDineroDisponibleBanco);

//Aumentamos los beneficios del grupo
//oBeneficios.setAumentoiEurosGrupo(iDineroDisponibleBanco);
}
```

```
public void mvAumentarDineroH(int iDinero) {
    int iNuevoDinero = cliente.getDinero() + iDinero;
    cliente.setDinero(iNuevoDinero);
    cliente.setNumeroVictorias(cliente.getNumeroVictorias() + 1);
}
```

```
public void mvDisminuirDineroH(int iDinero) {
    int iNuevoDinero = cliente.getDinero() - iDinero;
    cliente.setDinero(iNuevoDinero);
}
```

```
public String miApostarParlImpar() {
    try {
        boolean bPar = Random.class.newInstance().nextBoolean();
        //Quitamos el valor de la apuesta al hilo
        mvDisminuirDineroH(10);
        //Le damos el dinero al banco
        casino.setDinero(casino.getDinero() + 10);
        //Devolvemos el valor buleano aleatorio
        if (bPar) {
            return "true";
        } else {
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

return "false";
}

} catch (InstantiationException | IllegalAccessException ex) {
    Logger.getLogger(Cliente.class.getName()).log(Level.SEVERE, null, ex);
}

return "false";
}

public int miApostarMartinGala() {
    //Sacamos un numero al azar
    int iNumero = (int) (Math.random() * 36 + 1);
    //Quitamos el valor de la apuesta al hilo
    mvDisminuirDineroH(getDineroApuesta());
    //Le damos el dinero al banco
    casino.setDinero(casino.getDinero() + getDineroApuesta());

    return iNumero;
}

private void mvNumeroPerdidoMartinGala() {
    //oBeneficios.setAumentoEurosGrupo(getDineroApuesta());
    setDineroApuesta(getDineroApuesta() * 2);
}
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Paquete Controlador:

Controlador Abstracto:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ups.edu.ec.controlador;

import java.lang.reflect.ParameterizedType;
import java.util.List;
import javax.persistence.EntityManager;
import ups.edu.ec.JPAUtils.JPAUtils;


/**
 *
 * @author User
 * @param <E>
 */
public abstract class Controlador<E> {

    private EntityManager em;
    private Class<E> clase;

    public Controlador() {
        this.em = (EntityManager) JPAUtils.getEmf();

        java.lang.reflect.Type t = getClass().getGenericSuperclass();

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

ParameterizedType pt = (ParameterizedType) t;

clase = (Class) pt.getActualTypeArguments()[0];
}

public EntityManager getEm() {
    return em;
}

public void setEm(EntityManager em) {
    this.em = em;
}

public boolean create(E objeto) {


    try {

        em.getTransaction().begin();
        em.persist(objeto);
        em.getTransaction().commit();
        return true;
    } catch (Exception e) {
        e.printStackTrace();
        return false;
    }

}

public E read(int id) {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

return em.find(clase, id);
}

public boolean update(E objeto) {
    em.getTransaction().begin();
    em.merge(objeto);
    em.getTransaction().commit();
    return true;
}

public boolean delete(E objeto) {
    em.getTransaction().begin();
    if (!em.contains(objeto)) {
        objeto = em.merge(objeto);
    }

    em.remove(objeto);
    em.getTransaction().commit();
    return true;
}

public abstract List<E> findAll();

public abstract int getCodigo();
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Controlador Apuesta:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package ups.edu.ec.controlador;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import javax.persistence.Query;
import ups.edu.ec.modelo.Apuesta;

/**
 *
 * @author User
 */
public class ControladorApuesta extends Controlador<Apuesta> {

    public List<Apuesta> findByCodigoCliente(int codigo) {

        var listaTodos = findAll();
        List<Apuesta> lista = new ArrayList<>();

        listaTodos.stream().filter(ap -> (ap.getCodigoClienteFk().getCodigo() == codigo)).forEachOrdered(ap -> {
            lista.add(ap);
        });
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

return lista;

}

public List<Apuesta> findByNumeroApuesta(int codigo) {
    Query consulta = getEm().createNamedQuery("Apuesta.findByCodigo");
    consulta.setParameter("codigo", codigo);

    return consulta.getResultList();
}


public int numeroPartida() {
    var lista = findAll();

    if (lista.isEmpty()) {
        return 0;
    } else {
        Collections.sort(lista, (Apuesta ap1, Apuesta ap2) -> ap1.getId().compareTo(ap2.getId()));
        return lista.get(lista.size() - 1).getCodigo();
    }
}

@Override
public List<Apuesta> findAll() {
    Query consulta = getEm().createNamedQuery("Apuesta.findAll");
    return consulta.getResultList();
}

@Override

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public int getCodigo() {
    var lista = findAll();

    Collections.sort(lista, (Apuesta c1, Apuesta c2) -> c1.getCodigo().compareTo(c2.getCodigo()));

    return lista.get(lista.size()).getCodigo();
}

}

```

Controlador Cliente:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package ups.edu.ec.controlador;

import java.util.Collections;
import java.util.List;
import javax.persistence.Query;
import ups.edu.ec.modelo.Cliente;

/**
 *
 * @author User
 */
public class ControladorCliente extends Controlador<Cliente> {

    @Override

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public List<Cliente> findAll() {
    Query consulta = getEm().createNamedQuery("Cliente.findAll");
    return consulta.getResultList();
}

@Override
public int getCodigo() {
    var lista = findAll();
    Collections.sort(lista, (Cliente c1, Cliente c2) -> c1.getCodigo().compareTo(c2.getCodigo()));

    if (!lista.isEmpty()) {
        return lista.get(lista.size()-1).getCodigo();
    } else {
        return 0;
    }
}
}


```

Controlador Casino:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ups.edu.ec.controlador;

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

import java.util.List;

import javax.persistence.Query;

import ups.edu.ec.modelo.Casino;

/**
 *
 * @author User
 */
public class ControladorCasino extends Controlador<Casino> {

    @Override
    public List<Casino> findAll() {
        Query consulta = getEm().createNamedQuery("Casino.findAll");
        return consulta.getResultList();
    }

    @Override
    public int getCodigo() {
        return 1;
    }

}

```


Paquete vista:

Ventana General:

*

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

* and open the template in the editor.

*/

package ups.edu.ec.vista;

import ups.edu.ec.controlador.ControladorApuesta;

import ups.edu.ec.controlador.ControladorCasino;

import ups.edu.ec.controlador.ControladorCliente;

/**

*

* @author User

*/

public class VentanaGeneral extends javax.swing.JFrame {

private ControladorCliente controladorCliente;

private ControladorCasino controladorCasino;

private ControladorApuesta controladorApuesta;

private VentanaJuego ventanaJuego;

private VentanaJugadores ventanaJugadores;

private VentanaGestionApuestas ventanaGestionApuestas;

/**

* Creates new form VentanaGeneral

*/

public VentanaGeneral() {

initComponents();

controladorCliente = new ControladorCliente();

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

controladorCasino = new ControladorCasino();
controladorApuesta = new ControladorApuesta();

ventanaJuego = new VentanaJuego(controladorCliente, controladorCasino, controladorApuesta);
ventanaJugadores = new VentanaJugadores(controladorCliente, controladorApuesta, controladorCasino);
ventanaGestionApuestas = new VentanaGestionApuestas(controladorCliente, controladorApuesta);

desktopPane.add(ventanaJuego);
desktopPane.add(ventanaJugadores);
desktopPane.add(ventanaGestionApuestas);

}

public void cerrarVentanas() {
    ventanaJuego.hide();
    ventanaJugadores.hide();
    ventanaGestionApuestas.hide();
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    desktopPane = new javax.swing.JDesktopPane();

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

jLabel1 = new javax.swing.JLabel();

menuBar = new javax.swing.JMenuBar();

fileMenu = new javax.swing.JMenu();

juegoMenuItem = new javax.swing.JMenuItem();

jugadoresMenuItem = new javax.swing.JMenuItem();

apuestasMenuItem = new javax.swing.JMenuItem();

exitMenuItem = new javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ups/edu/ec/imagenes/ruletaPortada.jpg"))); //
NOI18N

desktopPane.add(jLabel1);

jLabel1.setBounds(0, 0, 690, 510);

fileMenu.setMnemonic('f');

fileMenu.setText("File");

juegoMenuItem.setMnemonic('o');

juegoMenuItem.setText("Juego");

juegoMenuItem.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        juegoMenuItemActionPerformed(evt);

    }

});

fileMenu.add(juegoMenuItem);

jugadoresMenuItem.setMnemonic('s');

jugadoresMenuItem.setText("Jugadores");

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

jugadoresMenuItem.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jugadoresMenuItemActionPerformed(evt);
    }
});

fileMenu.add(jugadoresMenuItem);

apuestasMenuItem.setMnemonic('a');
apuestasMenuItem.setText("Apuestas");
apuestasMenuItem.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        apuestasMenuItemActionPerformed(evt);
    }
});

fileMenu.add(apuestasMenuItem);

exitMenuItem.setMnemonic('x');
exitMenuItem.setText("Exit");
exitMenuItem.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        exitMenuItemActionPerformed(evt);
    }
});

fileMenu.add(exitMenuItem);

menuBar.add(fileMenu);

setJMenuBar(menuBar);

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(desktopPane, javax.swing.GroupLayout.DEFAULT_SIZE, 693, Short.MAX_VALUE)
);
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(desktopPane, javax.swing.GroupLayout.DEFAULT_SIZE, 510, Short.MAX_VALUE)
);

pack();
} // </editor-fold>

private void exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}

private void juegoMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    cerrarVentanas();
    ventanaJuego.setVisible(true);
}

private void jugadoresMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    cerrarVentanas();
    ventanaJugadores.setVisible(true);
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

}

private void apuestasMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    cerrarVentanas();
    ventanaGestionApuestas.setVisible(true);
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(VentanaGeneral.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    java.util.logging.Logger.getLogger(VentanaGeneral.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
} catch (IllegalAccessException ex) {
    java.util.logging.Logger.getLogger(VentanaGeneral.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
} catch (javax.swing.UnsupportedLookAndFeelException ex) {
    java.util.logging.Logger.getLogger(VentanaGeneral.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}
//</editor-fold>

```

```

/* Create and display the form */

```

```

java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new VentanaGeneral().setVisible(true);
    }
});
}

```

```

// Variables declaration - do not modify

```

```


private javax.swing.JMenuItem apuestasMenuItem;
private javax.swing.JDesktopPane desktopPane;
private javax.swing.JMenuItem exitMenuItem;
private javax.swing.JMenu fileMenu;
private javax.swing.JLabel jLabel1;
private javax.swing.JMenuItem juegoMenuItem;
private javax.swing.JMenuItem jugadoresMenuItem;
private javax.swing.JMenuBar menuBar;
// End of variables declaration

```

```

}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Ventana Juego:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ups.edu.ec.vista;

import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import ups.edu.ec.controlador.ControladorApuesta;
import ups.edu.ec.controlador.ControladorCasino;
import ups.edu.ec.controlador.ControladorCliente;
import ups.edu.ec.modelo.Apuesta;
import ups.edu.ec.modelo.Casino;
import ups.edu.ec.modelo.Cliente;
import ups.edu.ec.modelo.EnumTipoApuesta;
import ups.edu.ec.modelo.Hilo;

/**
 *
 * @author User

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

*/

```

public class VentanaJuego extends javax.swing.JInternalFrame {

    private ControladorCliente controladorCliente;
    private ControladorCasino controladorCasino;
    private ControladorApuesta controladorApuesta;

    private List<Cliente> listaNumeroConcreto;
    private List<Cliente> listaParImpar;
    private List<Cliente> listaMartinGala;
    private List<Cliente> listaJugadores;

    private List<Thread> listaThreads;

    private List<Hilo> listaHilos;
    private int iNumeroRuleta;

    private boolean juegoPausado;

    /**
     * Creates new form VentanaJuego
     *
     * @param controladorCliente
     * @param controladorCasino
     * @param controladorApuesta
     */
    public VentanaJuego(ControladorCliente controladorCliente, ControladorCasino controladorCasino,
        ControladorApuesta controladorApuesta) {
        initComponents();
    }

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

this.controladorApuesta = controladorApuesta;
this.controladorCasino = controladorCasino;
this.controladorCliente = controladorCliente;

listaMartinGala = new ArrayList<>();
listaNumeroConcreto = new ArrayList<>();
listaParImpar = new ArrayList<>();
listaJugadores = new ArrayList<>();
listaThreads = new ArrayList<>();
listaHilos = new ArrayList<>();

juegoPausado = false;
}


public String escogerJuego() {
    List<String> tiposJuego = new ArrayList<>();

    if (listaMartinGala.size() < 4) {
        tiposJuego.add(EnumTipoApuesta.MARTINGALA.toString());
    }

    if (listaNumeroConcreto.size() < 4) {
        tiposJuego.add(EnumTipoApuesta.NUMEROCONCRETO.toString());
    }

    if (listaParImpar.size() < 4) {
        tiposJuego.add(EnumTipoApuesta.PARIMPAR.toString());
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

Random randomico = new Random();

int eleccion = randomico.nextInt(3);

return tiposJuego.get(eleccion);

}

public void comenzarBotones() {
    btnComenzar.setEnabled(false);
    btnParar.setEnabled(true);
    btnTerminar.setEnabled(true);
}

public void agregarNumeroConcreto(Cliente cli) {

    listaNumeroConcreto.add(cli);
}

public void agregarParImpar(Cliente cli) {
    listaParImpar.add(cli);
}

public void agregarMartinGala(Cliente cli) {
    listaMartinGala.add(cli);
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

public void llenarTblClientes() {
    DefaultTableModel modelo = (DefaultTableModel) tblClientes.getModel();
    modelo.setRowCount(0);

    for (var cliente : listaJugadores) {
        Object[] row = {cliente.getCodigo(), cliente.getNombre().concat(cliente.getApellido()), cliente.getTipoApuesta()};
        modelo.addRow(row);
    }

    tblClientes.setModel(modelo);
}

public void limpiarCampos() {
    txtCodigo.setText("");
    llenarTblClientes();
}

public void limpiarListas() {
    listaJugadores.clear();
    listaMartinGala.clear();
    listaNumeroConcreto.clear();
    listaParImpar.clear();
    listaThreads.clear();
    btnJugadoresAleatorios.setEnabled(true);
}

public int getiNumeroRuleta() {
    return iNumeroRuleta;
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

}

public void setiNumeroRuleta(int iNumeroRuleta) {
    this.iNumeroRuleta = iNumeroRuleta;
}

public int miSacarNumero() {
    setiNumeroRuleta((int) (Math.random() * 36));
    return getiNumeroRuleta();
} //miSacarNumero()

public JTable getTblJuego() {
    return tblJuego;
}

public void setTblJuego(JTable tblJuego) {
    this.tblJuego = tblJuego;
}

public synchronized void llenarTablaJuego(List<Hilo> resultadoNumeroConcreto, List<Hilo> resultadoParImpar,
    List<Hilo> resultadoMartinGala) {

    DefaultTableModel modelo = (DefaultTableModel) tblJuego.getModel();
    //modelo.setRowCount(0);

    for (int i = 0; i < 4; i++) {
        Object[] row1 =
        {resultadoNumeroConcreto.get(i).getCliente().getNombre().concat(resultadoNumeroConcreto.get(i).getCliente().getApellido()),
            resultadoNumeroConcreto.get(i).getApuestas().get(0).getApostadoPara(),

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

resultadoNumeroConcreto.get(i).getApuestas().get(0).getDineroCliente(),
resultadoNumeroConcreto.get(i).getApuestas().get(0).getDineroCasino(),
resultadoNumeroConcreto.get(i).getCliente().getNumeroRuleta(),
resultadoNumeroConcreto.get(i).getCliente().getTipoApuesta());

```

```

modelo.addRow(row1);

```

```

String numeroEscogido;

```

```

if (resultadoParImpar.get(i).getApuestas().get(0).getApostadoPara().equalsIgnoreCase("true")) {
    numeroEscogido = "par";
} else {
    numeroEscogido = "impar";
}

```

```

Object[] row2 =
{resultadoParImpar.get(i).getCliente().getNombre().concat(resultadoParImpar.get(i).getCliente().getApellido()),
    numeroEscogido,
    resultadoParImpar.get(i).getApuestas().get(0).getDineroCliente(),
    resultadoParImpar.get(i).getApuestas().get(0).getDineroCasino(),
    resultadoParImpar.get(i).getCliente().getNumeroRuleta(),
    resultadoParImpar.get(i).getCliente().getTipoApuesta());

```

```

modelo.addRow(row2);

```

```

Object[] row3 =
{resultadoMartinGala.get(i).getCliente().getNombre().concat(resultadoMartinGala.get(i).getCliente().getApellido()),
    resultadoMartinGala.get(i).getApuestas().get(0).getApostadoPara(),
    resultadoMartinGala.get(i).getApuestas().get(0).getDineroCliente(),
    resultadoMartinGala.get(i).getApuestas().get(0).getDineroCasino(),
    resultadoMartinGala.get(i).getCliente().getNumeroRuleta(),
    resultadoMartinGala.get(i).getCliente().getTipoApuesta());

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        modelo.addRow(row3);

    }

    tblJuego.setModel(modelo);

}

public void llenarTabla(List<Hilo> hilos) {
    DefaultTableModel modelo = (DefaultTableModel) tblJuego.getModel();

    hilos.forEach(cl -> {
        cl.getApuestas().stream().map(ap -> {
            String numeroApostado = "";
            if (ap.getApostadoPara().equalsIgnoreCase("true")) {
                numeroApostado = "par";
            } else if ((ap.getApostadoPara().equalsIgnoreCase("false"))) {
                numeroApostado = "impar";
            } else {
                numeroApostado = ap.getApostadoPara();
            }
            ap.setApostadoPara(numeroApostado);
            Object[] row = {ap.getCodigoClienteFk().getNombre().concat(ap.getCodigoClienteFk().getApellido()),
                numeroApostado, ap.getDineroCliente(), ap.getDineroCasino(), ap.getResultadoRuleta(),
                ap.getTipoApuesta()};
            return row;
        }).forEachOrdered(row -> {
            modelo.addRow(row);
        });
    });
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

});

});

tblJuego.setModel(modelo);
}

public void guardarApuestas(List<Hilo> hilos) {

    hilos.forEach(hi -> {
        hi.getApuestas().forEach(ap -> {
            controladorApuesta.create(ap);
        });
    });
}

public void actualizarClientes(List<Hilo> hilos) {
    hilos.forEach(hi -> {
        hi.getApuestas().forEach(ap -> {
            controladorCliente.update(ap.getCodigoClienteFk());
        });
    });
}

public void actualizarCasino(List<Hilo> hilos) {
    var casino = hilos.get(hilos.size() - 1).getCasino();
    controladorCasino.update(casino);
}

/**

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

* This method is called from within the constructor to initialize the form.

* WARNING: Do NOT modify this code. The content of this method is always

* regenerated by the Form Editor.

*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

```
private void initComponents() {

    btnComenzar = new javax.swing.JButton();
    btnParar = new javax.swing.JButton();
    jPanel1 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    txtCodigo = new javax.swing.JTextField();
    btnAgregar = new javax.swing.JButton();
    cbxJuego = new javax.swing.JComboBox<>();
    jLabel2 = new javax.swing.JLabel();
    btnCancelar = new javax.swing.JButton();
    jScrollPane1 = new javax.swing.JScrollPane();
    tblClientes = new javax.swing.JTable();
    btnTerminar = new javax.swing.JButton();
    jPanel2 = new javax.swing.JPanel();
    jLabel3 = new javax.swing.JLabel();
    jPanel3 = new javax.swing.JPanel();
    jLabel4 = new javax.swing.JLabel();
    txtNumeroRuleta = new javax.swing.JTextField();
    jScrollPane2 = new javax.swing.JScrollPane();
    tblJuego = new javax.swing.JTable();
    btnJugadoresAleatorios = new javax.swing.JButton();
```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

btnComenzar.setText("Comenzar");

btnComenzar.setEnabled(false);

btnComenzar.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        btnComenzarActionPerformed(evt);

    }

});

```

```

btnParar.setText("Pausar");

btnParar.setEnabled(false);

btnParar.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        btnPararActionPerformed(evt);

    }

});

```

```

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(41, 43, 45)), "", javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP));

```

```

jLabel1.setText("Código:");

```

```

btnAgregar.setText("Agregar");

btnAgregar.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        btnAgregarActionPerformed(evt);

    }

});

```

```

cbxJuego.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "Cualquiera", "NumeroConcreto",
"ParImpar", "MartinGala" }));

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

jLabel2.setText("Juego:");

btnCancelar.setText("Cancelar");
btnCancelar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnCancelarActionPerformed(evt);
    }
});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .add(jPanel1Layout.createSequentialGroup()
                            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .add(jPanel1Layout.createSequentialGroup()
                                    .addContainerGap()
                                    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                        .addComponent(jLabel1)
                                        .addComponent(jLabel2))
                                    .addGap(18, 18, 18)
                                    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                                        .addComponent(cbxJuego, 0, 162, Short.MAX_VALUE)
                                        .addComponent(txtCodigo)))
                                    .addGroup(jPanel1Layout.createSequentialGroup()
                                        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                            .addGap(30, 30, 30)
                                            .addComponent(btnAgregar)
                                            .addGap(49, 49, 49)

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        .addComponent(btnCancelar)))

        .addContainerGap(19, Short.MAX_VALUE))

    );

    jPanel1Layout.setVerticalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addContainerGap()
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jLabel1)
                .addComponent(txtCodigo, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(18, 18, 18)
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(cbxJuego, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jLabel2))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(btnAgregar)
                .addComponent(btnCancelar))
            .addContainerGap(12, Short.MAX_VALUE))
        );

    tblClientes.setModel(new javax.swing.table.DefaultTableModel(
        new Object [][] {

        },
        new String [] {
            "Codigo", "Jugador", "Juego"
        }
    )

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

) {
    Class[] types = new Class [] {
        java.lang.Integer.class, java.lang.String.class, java.lang.String.class
    };
    boolean[] canEdit = new boolean [] {
        false, false, false
    };

    public Class getColumnClass(int columnIndex) {
        return types [columnIndex];
    }

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});

jScrollPane1.setViewportView(tblClientes);

btnTerminar.setText("Terminar");
btnTerminar.setEnabled(false);
btnTerminar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnTerminarActionPerformed(evt);
    }
});

jLabel3.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ups/edu/ec/imagenes/ruleta.gif"))); // NOI18N

javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout.setHorizontalGroup(
    jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jLabel3)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

jPanel2Layout.setVerticalGroup(
    jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 482,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

jLabel4.setText("Número Ruleta:");

txtNumeroRuleta.setEditable(false);

txtNumeroRuleta.setBackground(new java.awt.Color(153, 153, 153));

tblJuego.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {

    },
    new String [] {
        "Jugador", "Número", "Dinero", "Banco", "Ruleta", "Juego"
    }

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

) {
    Class[] types = new Class [] {
        java.lang.String.class, java.lang.Integer.class, java.lang.Integer.class, java.lang.Integer.class,
        java.lang.Integer.class, java.lang.String.class
    };
    boolean[] canEdit = new boolean [] {
        false, false, false, false, false, false
    };

    public Class getColumnClass(int columnIndex) {
        return types [columnIndex];
    }

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});

jScrollPane2.setViewportViewView(tblJuego);

javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
jPanel3.setLayout(jPanel3Layout);
jPanel3Layout.setHorizontalGroup(
    jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel3Layout.createSequentialGroup()
            .addGap(168, 168, 168)
            .addComponent(jLabel4)
            .addGap(59, 59, 59)
            .addComponent(txtNumeroRuleta, javax.swing.GroupLayout.PREFERRED_SIZE, 107,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(248, 248, 248)

.addComponent(btnJugadoresAleatorios)

.addGap(53, 53, 53)

.addComponent(btnComenzar)

.addGap(50, 50, 50)

.addComponent(btnParar, javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(47, 47, 47)

.addComponent(btnTerminar)

.addGap(0, 0, Short.MAX_VALUE))

.addGroup(layout.createSequentialGroup()

.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))

.addContainerGap())

.addGroup(layout.createSequentialGroup()

.addContainerGap(140, Short.MAX_VALUE)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(155, 155, 155)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(158, 158, 158))


);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

    .addComponent(btnComenzar)

    .addComponent(btnParar)

    .addComponent(btnTerminar)

    .addComponent(btnJugadoresAleatorios))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

    .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

    .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 0, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

    .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

    .addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))

.addContainerGap())

);

pack();
} // </editor-fold>

private void btnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    int codigo = Integer.valueOf(txtCodigo.getText());

    String opcionJuego = (String) cbxJuego.getSelectedItem();

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```


if (codigo == 0) {
    JOptionPane.showMessageDialog(this, "Llene todos los campos", "Error", JOptionPane.ERROR_MESSAGE);
} else {

    Cliente cliente = controladorCliente.read(codigo);
    if (cliente != null && !listaJugadores.contains(cliente)) {
        if (opcionJuego.equalsIgnoreCase("Cualquiera")) {
            opcionJuego = escogerJuego();
        }

        if (opcionJuego.equalsIgnoreCase("NumeroConcreto")) {
            if (listaNumeroConcreto.size() < 4) {
                cliente.setTipoApuesta(EnumTipoApuesta.NUMEROCONCRETO.toString());
                agregarNumeroConcreto(cliente);
            } else {
                JOptionPane.showMessageDialog(this, "Tipo de juego lleno", "Error", JOptionPane.ERROR_MESSAGE);
            }
        } else if (opcionJuego.equalsIgnoreCase("ParImpar")) {
            if (listaParImpar.size() < 4) {
                cliente.setTipoApuesta(EnumTipoApuesta.PARIMPAR.toString());
                agregarParImpar(cliente);
            } else {
                JOptionPane.showMessageDialog(this, "Tipo de juego lleno", "Error", JOptionPane.ERROR_MESSAGE);
            }
        } else {

            if (listaMartinGala.size() < 4) {
                cliente.setTipoApuesta(EnumTipoApuesta.MARTINGALA.toString());
            }
        }
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        agregarMartinGala(cliente);
    } else {
        JOptionPane.showMessageDialog(this, "Tipo de juego lleno", "Error", JOptionPane.ERROR_MESSAGE);
    }
}

listaJugadores.add(cliente);
limpiarCampos();
btnJugadoresAleatorios.setEnabled(false);
if (listaJugadores.size() == 12) {
    btnAgregar.setEnabled(false);
    btnComenzar.setEnabled(true);
}

JOptionPane.showMessageDialog(this, "Jugador agregado con exito", "Mensaje",
JOptionPane.INFORMATION_MESSAGE);
} else {
    JOptionPane.showMessageDialog(this, "Jugador no encontrado", "Error", JOptionPane.ERROR_MESSAGE);
}
}

}

private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    limpiarListas();
    limpiarCampos();
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```
private void btnJugadoresAleatoriosActionPerformed(java.awt.event.ActionEvent evt) {
```

```
// TODO add your handling code here:
```

```
int tamanoJugadores = controladorCliente.findAll().size();
```

```
int numeroJugadores = 4;
```

```
var jugadores = controladorCliente.findAll();
```

```
Random numeroRandom = new Random();
```

```
for (int i = 0; i < numeroJugadores; i++) {
```

```
    int posicion = numeroRandom.nextInt(tamanoJugadores);
```

```
    var cl = jugadores.get(posicion);
```

```
    if (!listaJugadores.contains(cl)) {
```

```
        cl.setTipoApuesta(EnumTipoApuesta.NUMEROCONCRETO.toString());
```

```
        listaJugadores.add(cl);
```

```
        listaNumeroConcreto.add(cl);
```

```
    } else {
```

```
        numeroJugadores++;
```

```
    }
```

```
}
```

```
numeroJugadores = 4;
```

```
for (int i = 0; i < numeroJugadores; i++) {
```


```
    int posicion = numeroRandom.nextInt(tamanoJugadores);
```

```
    var cl = jugadores.get(posicion);
```

```
    if (!listaJugadores.contains(cl)) {
```

```
        cl.setTipoApuesta(EnumTipoApuesta.PARIMPAR.toString());
```

```
        listaJugadores.add(cl);
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        listaParImpar.add(cl);
    } else {
        numeroJugadores++;
    }
}

numeroJugadores = 4;
for (int i = 0; i < numeroJugadores; i++) {
    int posicion = numeroRandom.nextInt(tamanoJugadores);
    var cl = jugadores.get(posicion);
    if (!listaJugadores.contains(cl)) {
        cl.setTipoApuesta(EnumTipoApuesta.MARTINGALA.toString());
        listaJugadores.add(cl);
        listaMartinGala.add(cl);
    } else {
        numeroJugadores++;
    }
}

limpiarCampos();
btnJugadoresAleatorios.setEnabled(false);
btnComenzar.setEnabled(true);

}

private void btnComenzarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

comenzarBotones();

var oBanco = controladorCasino.findAll().get(0);

int numeroRuleta = miSacarNumero();

boolean bSeguir = true;
int iNumeroPartida = controladorApuesta.numeroPartida() + 1;

int contador = 0;

List<Hilo> listaHilosNumeroConceto = new ArrayList<>();
List<Hilo> listaHilosParImpar = new ArrayList<>();
List<Hilo> listaHilosMartinGala = new ArrayList<>();

for (int i = 0; i < 4; i++) {
    listaHilosNumeroConceto.add(new Hilo(listaNumeroConcreto.get(i),
        oBanco, iNumeroPartida, controladorCliente, controladorApuesta, controladorCasino, this));

    listaHilosParImpar.add(new Hilo(listaParImpar.get(i),
        oBanco, iNumeroPartida, controladorCliente, controladorApuesta, controladorCasino, this));

    listaHilosMartinGala.add(new Hilo(listaMartinGala.get(i),
        oBanco, iNumeroPartida, controladorCliente, controladorApuesta, controladorCasino, this));
}

while (bSeguir) {
    System.out.println(iNumeroPartida + "----- Numero Ruleta: " + iNumeroRuleta);

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

if (iNumeroRuleta != 0 && contador != 2) {
    try {
        for (int i = 0; i < 4; i++) {
            System.out.println("hhhhhhhhhhhhhhhhhhhhhhhhhhhhhh");
            //Le pasamos el valor que ha sacado el crupier a los hilos
            listaNumeroConcreto.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));
            listaParImpar.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));
            listaMartinGala.get(i).setNumeroRuleta(String.valueOf(iNumeroRuleta));

            listaHilosNumeroConceto.get(i).setNumeroPartida(iNumeroPartida);
            listaHilosParImpar.get(i).setNumeroPartida(iNumeroPartida);
            listaHilosMartinGala.get(i).setNumeroPartida(iNumeroPartida);
            listaHilos.add(listaHilosNumeroConceto.get(i));
            listaHilos.add(listaHilosParImpar.get(i));
            listaHilos.add(listaHilosMartinGala.get(i));

            //hilo3.setDineroApuesta(10);
            Thread th = new Thread(listaHilosNumeroConceto.get(i));
            Thread thPI = new Thread(listaHilosParImpar.get(i));
            Thread thMa = new Thread(listaHilosMartinGala.get(i));

            th.setName("Hilo " + i + " NCo");
            thPI.setName("Hilo " + i + " PI");
            thMa.setName("Hilo " + i + " MA");
            listaThreads.add(th);
            listaThreads.add(thPI);
            listaThreads.add(thMa);

            //llenarTablaJuego(listaHilosNumeroConceto, listaHilosParImpar, listaHilosMartinGala);

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

//Inicializamos los hilos

th.start();

thPI.start();

thMa.start();

//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParImpar, listaHilosMartinGala);
} // for()

iNumeroPartida++;

txtNumeroRuleta.setText(String.valueOf(iNumeroRuleta));

//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParImpar, listaHilosMartinGala);

//Llamamos al metodo sleep para que se imprima la cuantia del banco justo despues de que se ejecuten los hilos
Thread.sleep(50);

System.out.println("BANCO: " + oBanco.getDinero());


//Volvemos a utilizar el metodo sleep como se pide en el enunciado
Thread.sleep(3000);

//Sacamos otro numero pasados esos 3 segundos

iNumeroRuleta = miSacarNumero();

contador++;

} catch (InterruptedException ex) {

    Logger.getLogger(NewMain.class.getName()).log(Level.SEVERE, null, ex);

}

} else {

    bSeguir = false;


}

}

llenarTabla(listaHilos);

//llenarTablaJuego(listaHilosNumeroConceto, listaHilosParImpar, listaHilosMartinGala);

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

}

private void btnPararActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    if (!juegoPausado) {
        listaThreads.forEach(hi -> {
            hi.interrupt();
        });
    } else {
        listaThreads.forEach(Thread::resume);
    }
}

private void btnTerminarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    try {
        listaThreads.forEach(hi -> {
            hi.stop();
        });

        guardarApuestas(listaHilos);
        actualizarClientes(listaHilos);
        actualizarCasino(listaHilos);

        System.out.println(listaJugadores);
        btnComenzar.setEnabled(false);
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        btnParar.setEnabled(false);

        btnComenzar.setEnabled(false);

        btnJugadoresAleatorios.setEnabled(true);

        btnTerminar.setEnabled(false);
    } catch (Exception e) {
        e.printStackTrace();
    }

    //llenarTabla(listaHilos);

}

// Variables declaration - do not modify
private javax.swing.JButton btnAgregar;
private javax.swing.JButton btnCancelar;
private javax.swing.JButton btnComenzar;
private javax.swing.JButton btnJugadoresAleatorios;
private javax.swing.JButton btnParar;
private javax.swing.JButton btnTerminar;
private javax.swing.JComboBox<String> cbxJuego;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JPanel jPanel3;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JScrollPane jScrollPane2;

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```
private javax.swing.JTable tblClientes;

private javax.swing.JTable tblJuego;

private javax.swing.JTextField txtCodigo;

private javax.swing.JTextField txtNumeroRuleta;

// End of variables declaration

}
```

Ventana Jugadores:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package ups.edu.ec.vista;

import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import ups.edu.ec.controlador.ControladorApuesta;
import ups.edu.ec.controlador.ControladorCasino;
import ups.edu.ec.controlador.ControladorCliente;
import ups.edu.ec.modelo.Cliente;

/**
 *
 * @author User
 */
public class VentanaJugadores extends javax.swing.JInternalFrame {
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

private ControladorCliente controladorCliente;

private ControladorApuesta controladorApuesta;

private ControladorCasino controladorCasino;


private Cliente cl;


/**
 * Creates new form VentanaJugadores
 *
 * @param controladorCliente
 * @param controladorApuesta
 * @param controladorCasino
 */

public VentanaJugadores(ControladorCliente controladorCliente, ControladorApuesta controladorApuesta,
    ControladorCasino controladorCasino) {
    initComponents();

    this.controladorCliente = controladorCliente;
    this.controladorCasino = controladorCasino;
    this.controladorApuesta = controladorApuesta;


    limpiar();
}

public void limpiar() {
    llenarTblClientes();
    cargarCodigo();
    txtNombre.setText("");
    txtApellido.setText("");
    txtDinero.setText("1000");

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

txtNumeroVictorias.setText("");
txtNumeroDerrotas.setText("");
txtDinero.setEditable(false);
botonesInicio();

}

public void botonesInicio() {
    btnCrear.setEnabled(true);
    btnCancelar.setEnabled(true);
    btnActualizar.setEnabled(false);
    btnEliminar.setEnabled(false);
}

public void botonesActualizar() {
    btnCrear.setEnabled(false);
    btnCancelar.setEnabled(true);
    btnActualizar.setEnabled(true);
    btnEliminar.setEnabled(true);
}

public void cargarCodigo() {
    txtCodigo.setText(String.valueOf(controladorCliente.getCodigo() + 1));
}

public void llenarTblClientes() {
    var lista = controladorCliente.findAll();

    DefaultTableModel modelo = (DefaultTableModel) tblClientes.getModel();

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

modelo.setRowCount(0);

for (Cliente cliente : lista) {
    Object[] row = {cliente.getCodigo(), cliente.getNombre(), cliente.getApellido(), cliente.getDinero()};
    modelo.addRow(row);
}

tblClientes.setModel(modelo);

}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    jLabel2 = new javax.swing.JLabel();
    jLabel3 = new javax.swing.JLabel();
    jLabel4 = new javax.swing.JLabel();
    jLabel5 = new javax.swing.JLabel();
    jLabel6 = new javax.swing.JLabel();
    btnCrear = new javax.swing.JButton();
    btnActualizar = new javax.swing.JButton();

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

btnEliminar = new javax.swing.JButton();
btnCancelar = new javax.swing.JButton();
txtCodigo = new javax.swing.JTextField();
txtNombre = new javax.swing.JTextField();
txtApellido = new javax.swing.JTextField();
txtDinero = new javax.swing.JTextField();
txtNumeroVictorias = new javax.swing.JTextField();
txtNumeroDerrotas = new javax.swing.JTextField();
jScrollPane1 = new javax.swing.JScrollPane();
tblClientes = new javax.swing.JTable();

setClosable(true);
setDefaultCloseOperation(javax.swing.WindowConstants.HIDE_ON_CLOSE);

jLabel1.setText("Código:");

jLabel2.setText("Nombre:");

jLabel3.setText("Apellido:");


jLabel4.setText("Dinero:");

jLabel5.setText("Número victorias:");

jLabel6.setText("Número derrotas:");

btnCrear.setText("Crear");
btnCrear.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    btnCrearActionPerformed(evt);
}
});

btnActualizar.setText("Actualizar");
btnActualizar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnActualizarActionPerformed(evt);
    }
});

btnEliminar.setText("Eliminar");
btnEliminar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnEliminarActionPerformed(evt);
    }
});

btnCancelar.setText("Cancelar");
btnCancelar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnCancelarActionPerformed(evt);
    }
});

txtCodigo.setEditable(false);
txtCodigo.setBackground(new java.awt.Color(102, 102, 102));

txtDinero.setEditable(false);

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        .addComponent(txtDinero, javax.swing.GroupLayout.DEFAULT_SIZE, 180, Short.MAX_VALUE)

        .addComponent(txtNumeroVictorias)

        .addComponent(txtNumeroDerrotas))

        .addGap(12, 12, 12))

    .addGroup(jPanel1Layout.createSequentialGroup())

        .addGap(39, 39, 39)

        .addComponent(btnCrear)

        .addGap(27, 27, 27)

        .addComponent(btnActualizar)

        .addGap(38, 38, 38)

        .addComponent(btnEliminar)

        .addGap(28, 28, 28)

        .addComponent(btnCancelar)))

    .addContainerGap(41, Short.MAX_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup())

    .addGap(26, 26, 26)

    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

        .addComponent(jLabel1)

        .addComponent(txtCodigo, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

    .addGap(18, 18, 18)

    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)


        .addComponent(jLabel2)

        .addComponent(txtNombre, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

    .addGap(18, 18, 18)

    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

.addComponent(jLabel3)

.addComponent(txtApellido, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel4)

.addComponent(txtDinero, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel5)

.addComponent(txtNumeroVictorias, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel6)

.addComponent(txtNumeroDerrotas, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addGap(27, 27, 27)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(btnCrear)

.addComponent(btnActualizar)

.addComponent(btnEliminar)

.addComponent(btnCancelar))

.addContainerGap(24, Short.MAX_VALUE))

);

tblClientes.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

},
new String [] {
    "Código", "Nombre", "Apellido", "Dinero"
}
){
    Class[] types = new Class [] {
        java.lang.Integer.class, java.lang.String.class, java.lang.String.class, java.lang.Integer.class
    };
    boolean[] canEdit = new boolean [] {
        false, false, false, false
    };

    public Class getColumnClass(int columnIndex) {
        return types [columnIndex];
    }

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});

tblClientes.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        tblClientesMouseClicked(evt);
    }
});

jScrollPane1.setViewportView(tblClientes);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```


layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(31, 31, 31)
            .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 526,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(20, Short.MAX_VALUE))
        );
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(0, 26, Short.MAX_VALUE))
        );

pack();
} // </editor-fold>

private void btnCrearActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    String nombre = txtNombre.getText();
    String apellido = txtApellido.getText();

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

if (nombre.isBlank() || apellido.isBlank()) {
    JOptionPane.showMessageDialog(this, "Llene todos los campos", "Error", JOptionPane.ERROR_MESSAGE);
} else {

    Cliente cliente = new Cliente(nombre, apellido);

    if (controladorCliente.create(cliente)) {
        JOptionPane.showMessageDialog(this, "Cliente creada con exito", "Mensaje",
JOptionPane.INFORMATION_MESSAGE);
        limpiar();
    } else {
        JOptionPane.showMessageDialog(this, "Cliente ya existente", "Error", JOptionPane.ERROR_MESSAGE);
    }
}

}


private void tblClientesMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:

    int fila = tblClientes.getSelectedRow();
    int codigo = (int) tblClientes.getValueAt(fila, 0);

    cl = controladorCliente.read(codigo);

    txtCodigo.setText(String.valueOf(cl.getCodigo()));
    txtNombre.setText(cl.getNombre());
    txtApellido.setText(cl.getApellido());

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

txtDinero.setText(String.valueOf(cl.getDinero()));

if (cl.getDinero() < 1000) {
    txtDinero.setEditable(true);
}

txtNumeroVictorias.setText(String.valueOf(cl.getNumeroVictorias()));
txtNumeroDerrotas.setText(String.valueOf(cl.getNumeroDerrotas()));

botonesActualizar();
}

private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    String nombre = txtNombre.getText();
    String apellido = txtApellido.getText();
    int dinero = 1000;
    if (txtDinero.isEditable() || dinero <= 1000) {
        dinero = Integer.valueOf(txtDinero.getText());
    }

    if (nombre.isBlank() || apellido.isBlank() || dinero > 1000) {
        JOptionPane.showMessageDialog(this, "Llene todos los campos, dinero maximo $1000", "Error",
        JOptionPane.ERROR_MESSAGE);
    } else {

        cl.setNombre(nombre);
        cl.setApellido(apellido);
        cl.setDinero(dinero);
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    if (controladorCliente.create(cl)) {
        JOptionPane.showMessageDialog(this, "Cliente actualizado con exito", "Mensaje",
JOptionPane.INFORMATION_MESSAGE);
        limpiar();
    } else {
        JOptionPane.showMessageDialog(this, "Ha ocurrido un error", "Error", JOptionPane.ERROR_MESSAGE);
    }
}

}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    Cliente cliente = controladorCliente.read(Integer.valueOf(txtCodigo.getText()));

    if (controladorCliente.delete(cliente)) {

        JOptionPane.showMessageDialog(this, "Casa eliminada con exito", "Mensaje",
JOptionPane.INFORMATION_MESSAGE);
        limpiar();
    } else {
        JOptionPane.showMessageDialog(this, "Ha ocurrido un error", "Error", JOptionPane.ERROR_MESSAGE);
    }

}

private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```
limpiar();
```

```
}
```

```
// Variables declaration - do not modify
```

```
private javax.swing.JButton btnActualizar;
```

```
private javax.swing.JButton btnCancelar;
```

```
private javax.swing.JButton btnCrear;
```

```
private javax.swing.JButton btnEliminar;
```

```
private javax.swing.JLabel jLabel1;
```

```
private javax.swing.JLabel jLabel2;
```

```
private javax.swing.JLabel jLabel3;
```

```
private javax.swing.JLabel jLabel4;
```

```
private javax.swing.JLabel jLabel5;
```

```
private javax.swing.JLabel jLabel6;
```

```
private javax.swing.JPanel jPanel1;
```

```
private javax.swing.JScrollPane jScrollPane1;
```

```
private javax.swing.JTable tblClientes;
```

```
private javax.swing.JTextField txtApellido;
```

```
private javax.swing.JTextField txtCodigo;
```

```
private javax.swing.JTextField txtDinero;
```


```
private javax.swing.JTextField txtNombre;
```

```
private javax.swing.JTextField txtNumeroDerrotas;
```

```
private javax.swing.JTextField txtNumeroVictorias;
```

```
// End of variables declaration
```

```
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Ventana Gestión Apuestas:

/*

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

* and open the template in the editor.

*/

package ups.edu.ec.vista;

import java.util.List;

import javax.swing.table.DefaultTableModel;

import ups.edu.ec.controlador.ControladorApuesta;

import ups.edu.ec.controlador.ControladorCliente;

import ups.edu.ec.modelo.Apuesta;

/**

*

* @author User

*/

public class VentanaGestionApuestas extends javax.swing.JInternalFrame {

private ControladorCliente controladorCliente;

private ControladorApuesta controladorApuesta;

/**

* Creates new form VentanaGestionApuestas

*

* @param controladorCliente

* @param controladorApuesta

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

*/

```
public VentanaGestionApuestas(ControladorCliente controladorCliente,
```

```
    ControladorApuesta controladorApuesta) {
```

```
    initComponents();
```

```
    this.controladorApuesta = controladorApuesta;
```

```
    this.controladorCliente = controladorCliente;
```

```
}
```

```
public void limpiar() {
```

```
    txtBusqueda.setText("");
```

```
    txtBusqueda.setEditable(false);
```

```
    DefaultTableModel modelo = (DefaultTableModel) tblApuestas.getModel();
```

```
    modelo.setRowCount(0);
```

```
    cbxBusqueda.setSelectedIndex(0);
```

```
    btnBuscar.setEnabled(false);
```

```
}
```

```
public void llenarTbl(List<Apuesta> apuestas) {
```

```
    DefaultTableModel modelo = (DefaultTableModel) tblApuestas.getModel();
```

```
    modelo.setRowCount(0);
```

```
    for (Apuesta ap : apuestas) {
```

```
        Object[] row = {ap.getId(), ap.getCodigo(),
```

```
        ap.getCodigoClienteFk().getNombre().concat(ap.getCodigoClienteFk().getApellido()),
```

```
        ap.getApostadoPara(), ap.getResultadoRuleta(), ap.getCantidadApuesta(),
```

```
        ap.getDineroCliente(), ap.getDineroCasino(), ap.getGanador(), ap.getTipoApuesta());
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    modelo.addRow(row);
}

tblApuestas.setModel(modelo);

}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    cbxBusqueda = new javax.swing.JComboBox<>();
    txtBusqueda = new javax.swing.JTextField();
    btnBuscar = new javax.swing.JButton();
    btnCancelar = new javax.swing.JButton();
    jScrollPane1 = new javax.swing.JScrollPane();
    tblApuestas = new javax.swing.JTable();

    jLabel1.setText("Busqueda:");

    cbxBusqueda.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "--Seleccione--", "CodigoCliente",
"NumeroApuesta" }));

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

cbxBusqueda.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        cbxBusquedaActionPerformed(evt);
    }
});

txtBusqueda.setEditable(false);

btnBuscar.setText("Buscar");
btnBuscar.setEnabled(false);
btnBuscar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnBuscarActionPerformed(evt);
    }
});

btnCancelar.setText("Cancelar");
btnCancelar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnCancelarActionPerformed(evt);
    }
});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(30, 30, 30)

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(cbxBusqueda, javax.swing.GroupLayout.PREFERRED_SIZE, 140,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addComponent(jLabel1))
.addGap(27, 27, 27)
.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
    .addComponent(txtBusqueda, javax.swing.GroupLayout.PREFERRED_SIZE, 201,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGroup(jPanel1Layout.createSequentialGroup()
        .addComponent(btnBuscar, javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(28, 28, 28)
        .addComponent(btnCancelar, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
    .addContainerGap(41, Short.MAX_VALUE))
);
jPanel1Layout.setVerticalGroup(
jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel1Layout.createSequentialGroup()
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel1)
            .addComponent(txtBusqueda, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(18, 18, 18)
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(cbxBusqueda, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(btnBuscar)
            .addComponent(btnCancelar))
        .addContainerGap(20, Short.MAX_VALUE))

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

);

tblApuestas.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {

    },
    new String [] {
        "Código", "Partida", "Jugador", "Numero", "Ruleta", "Dinero Apuesta", "Dinero jugador", "Dinero casino", "Ganador",
        "Tipo"
    }
) {
    Class[] types = new Class [] {
        java.lang.Integer.class, java.lang.Integer.class, java.lang.String.class, java.lang.String.class, java.lang.String.class,
        java.lang.Integer.class, java.lang.Integer.class, java.lang.Integer.class, java.lang.String.class, java.lang.Object.class
    };

    boolean[] canEdit = new boolean [] {
        false, false, false, false, false, false, false, false, false, false
    };

    public Class getColumnClass(int columnIndex) {
        return types [columnIndex];
    }

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});

jScrollPane1.setViewportViewView(tblApuestas);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

getContentPane().setLayout(layout);

layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(300, 300, 300)
            .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(253, Short.MAX_VALUE))
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jScrollPane1))
    );

layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
                javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
    );

pack();
} // </editor-fold>

private void cbxBusquedaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    String seleccion = (String) cbxBusqueda.getSelectedItem();

```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

if (seleccion.equalsIgnoreCase("CodigoCliente")) {
    btnBuscar.setEnabled(true);
    txtBusqueda.setEditable(true);
} else if (seleccion.equalsIgnoreCase("NumeroApuesta")) {
    btnBuscar.setEnabled(true);
    txtBusqueda.setEditable(true);
} else {
    btnBuscar.setEnabled(false);
    txtBusqueda.setEditable(false);
}

}

private void btnBuscarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    String seleccion = (String) cbxBusqueda.getSelectedItem();
    int codigo = Integer.valueOf(txtBusqueda.getText());

    if (seleccion.equalsIgnoreCase("CodigoCliente")) {
        var cliente = controladorCliente.read(codigo);

        if (cliente != null) {
            var lista = controladorApuesta.findByCodigoCliente(codigo);
            llenarTbl(lista);
        }

    } else {

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```
var lista = controladorApuesta.findByNumeroApuesta(codigo);
```

```
llenarTbl(lista);
```

```
}
```

```
}
```

```
private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
```

```
// TODO add your handling code here:
```

```
limpiar();
```

```
}
```

```
// Variables declaration - do not modify
```

```
private javax.swing.JButton btnBuscar;
```

```
private javax.swing.JButton btnCancelar;
```

```
private javax.swing.JComboBox<String> cbxBusqueda;
```

```
private javax.swing.JLabel jLabel1;
```

```
private javax.swing.JPanel jPanel1;
```

```
private javax.swing.JScrollPane jScrollPane1;
```

```
private javax.swing.JTable tblApuestas;
```

```
private javax.swing.JTextField txtBusqueda;
```

```
// End of variables declaration
```

```
}
```

 UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

RESULTADO(S) OBTENIDO(S):

- Lograr afianzar los conocimientos sobre Threads y JPA.
- Crear una aplicación donde existen varios hilos al mismo tiempo.
- Persistencia de datos a través de JPA.

CONCLUSIONES:

En conclusión, esta práctica ha sido de mucha ayuda para practicar lo que se ha aprendido a lo largo de este ciclo, y además, está nos sirve para obtener un conocimiento base sobre como funcionan los videojuegos ya que ahí es donde más se utilizan hilos.

RECOMENDACIONES:

No existe ninguna recomendación.

Nombre de estudiante: Adolfo Sebastián Jara Gavilanes.

Firma de estudiante:

