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

Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación

Universidad Politécnica Salesiana

Vicerrectorado Docente

Código del Formato:	GUIA-PRL-001
Versión:	VF1.0
Elaborado por:	Directores de Área del Conocimiento Integrantes Consejo Académico
Fecha de elaboración:	2016/04/01
Revisado por:	Consejo Académico
Fecha de revisión:	2016/04/06
Aprobado por:	Lauro Fernando Pesántez Avilés Vicerrector Docente
Fecha de aprobación:	2016/14/06
Nivel de confidencialidad:	Interno

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

Descripción General

Propósito


El propósito del presente documento es definir un estándar para elaborar documentación de guías de práctica de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana, con la finalidad de lograr una homogenización en la presentación de la información por parte del personal académico y técnico docente.


Alcance


El presente estándar será aplicado a toda la documentación referente a informes de prácticas de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana.

Formatos

- Formato de Guía de Práctica de Laboratorio / Talleres / Centros de Simulación – para Docentes
- Formato de Informe de Práctica de Laboratorio / Talleres / Centros de Simulación – para Estudiantes

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

		FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES	
CARRERA: COMPUTACIÓN		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Examen Practico Java	
OBJETIVO: Identificar los cambios importantes de Java Diseñar e Implementar expresiones regulares Entender la cada uno de las características nuevas en Java			
INSTRUCCIONES (Detallar las instrucciones que se dará al estudiante):		1. Revisar los conceptos fundamentales de Java	
		2. Establecer las características de Java en programación genérica	
		3. Implementar y diseñar los nuevos componentes de programación genérica	
		4. Realizar el informe respectivo según los datos solicitados.	
ACTIVIDADES POR DESARROLLAR (Anotar las actividades que deberá seguir el estudiante para el cumplimiento de la práctica)			
1. Revisar la teoría y conceptos de Java 8, 9 ,10, 11, 12			
2. Diseñar e implementar las características de Java para generar una expresion regular.			
3. Probar su funcionamiento y rendimiento dentro de los equipos de cómputo de programación genérica.			
4. Realizar práctica codificando los codigos de las nuevas características de Java y su uso dentro de un sistema escolar.			
Enunciado Se desea generar un sistema que me permita extraer infomación del internet a traves de expresiones regulares, esta informacion permitira vincular actividades desarrolladas del los niños con aplicaciones mobiles que permitan apoyar en el desarrollo de las actividades planteadas (https://play.google.com/store?hl=es&gl=US). Adicionalmente, se debe realizar un sistema de gestion de alumnos y actividades planificadas por curso, dentro de este sistema se debe realizar un procesos de administracion de usuarios los mismo que son los docentes de cada curso escolar, en este sentido solo debemos tener un administrador (Rector) el encargado de crear docentes y el curso que se le asigna. Ejemplo Rector: Docentes: 1. Diego Quisi 2. Vladimir Robles 3. Etc. Cursos: 1 de basica 2 de basica 3 basica			

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

Asignacion de Curso – Docente

1 Basica -> Diego Quisi

2 Basica -> Vladimir Robles

Dentro de cada curso el docente gestionara los estudiantes y las actividades planificadas para el curso, estas actividades tendra una opcion de buscar aplicaciones moviles dentro de la tiendas de play store, obtenidas desde el internet, dentro de esta información lo importante es mostrar el link y una descripción para ello deberán utilizar expresiones regulares.

Ejemplo Docentes:

Alumnos

1. Juan Perez

2. Maria Peralta

3. .

Actividades:

1. Suma de numeros -> Obtener aplicaciones moviles (Link y Titulo)
2. Resta de numeros -> Obtener aplicaciones moviles
3. Oraciones compuestas -> Obtener aplicaciones moviles
4. Etc.

Toda esta infomación sera almacenada dentro de archivos y deberan tener aplicado al menos una patron de diseño y las nuevas características de programación de Java 8 o superior.

Al finalizar, generar el informe de la practica en formato PDF y subir todo el proyecto incluido el informe al repositorio personal.

La fecha de entrega: 23:55 del 01 de diciembre del 2020.

RESULTADO(S) OBTENIDO(S):

Realizar procesos de investigación sobre los cambios importantes de Java

Entender las aplicaciones de codificación de las nuevas características en base a la programación genérica y expresiones regulares.

Entender las funcionalidades adicionales de Java.

CONCLUSIONES:


Aprenden a trabajar en grupo dentro de plazos de tiempo establecidos, manejando el lenguaje de programación de Java.


RECOMENDACIONES:

Realizar el trabajo dentro del tiempo establecido.

Docente / Técnico Docente: _____

Firma: _____

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

		FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES	
CARRERA: Computación		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Examen Practico Java	
OBJETIVO ALCANZADO: <ul style="list-style-type: none"> • Identificar los cambios importantes de Java. • Diseñar e Implementar expresiones regulares. • Entender la cada uno de las características nuevas en Java. 			
ACTIVIDADES DESARROLLADAS			
1. Revisar la teoría y conceptos de Java 8, 9 ,10, 11, 12			
2. Diseñar e implementar las características de Java para generar una expresión regular.			
3. Probar su funcionamiento y rendimiento dentro de los equipos de cómputo de programación genérica.			
4. Realizar práctica codificando los códigos de las nuevas características de Java y su uso dentro de un sistema escolar.			
PROGRAMA DE GESTION DE ACTIVIDADES DE UN COLEGIO Paquetes con sus respectivas clases. -.ec.edu.ups.controlador <ul style="list-style-type: none"> ➤ Abstract Controlador <pre>package ec.edu.ups.controlador; import java.io.File; import java.io.FileInputStream; import java.io.FileOutputStream; import java.io.IOException; import java.io.ObjectInputStream; import java.io.ObjectOutputStream; import java.util.ArrayList; import java.util.List; import java.util.Optional; /** * * @author paul_ * @param <E> */ public abstract class AbstractControlador<E> { private String ruta; private List<E> lista; public AbstractControlador(String ruta) { lista = new ArrayList(); this.ruta = ruta; } }</pre>			

```
public void cargarDatos() throws ClassNotFoundException, IOException {
    ObjectInputStream datos = null;
    try {
        File f = new File(ruta);
        FileInputStream a = new FileInputStream(f);
        datos = new ObjectInputStream(a);

        lista = (List<E>) datos.readObject();

    } catch (IOException e) {

    }
}

public void guardarDatos(String ruta) throws IOException {
    ObjectOutputStream datos = null;
    File f = new File(ruta);
    FileOutputStream archivo = new FileOutputStream(f);
    datos = new ObjectOutputStream(archivo);
    datos.writeObject(lista);
}

public boolean crear(E objeto) {

    if (validar(objeto) == true) {
        return lista.add(objeto);
    }
    return false;
}

public Optional<E> buscar(E comparar) {
    return lista.stream().filter(objeto -> objeto.equals(comparar)).findFirst();
}

public int posicion(E objetoC) {
    for (int i = 0; i < lista.size(); i++) {
        E objetoL = lista.get(i);
        if (objetoL.equals(objetoC)) {
            return i;
        }
    }
    return -1;
}

public boolean eliminar(E objeto) {
    Optional<E> buscar = buscar(objeto);
    E objetoE = buscar.get();
    if (objetoE != null) {
        System.out.println("Verdadero");
    }
}
```

```
        return lista.remove(objetoE);

    }
    System.out.println("Falso");
    return false;

}

public boolean actualizar(E objetoA) {
    int pos = posicion(objetoA);
    if (pos >= 0) {
        lista.set(pos, objetoA);
        System.out.println("TRUE");
        return true;
    }
    System.out.println("FALSE");
    return false;
}

public abstract boolean validar(E objeto);

public abstract int generarId();

public List<E> getLista() {
    return lista;
}


public void setLista(List<E> lista) {
    this.lista = lista;
}
}

➤ Controlador Actividad
package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Actividad;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.Iterator;
import java.util.List;
import java.util.Set;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 *
 * @author paul_
 */
public class ControladorActividad extends AbstractControlador<Actividad>{

    private Pattern patron;
```

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

```

private Matcher corpus;

public ControladorActividad(String ruta) {
    super(ruta);
}

@Override
public boolean validar(Actividad objeto) {
    return true;
}

@Override
public int generarId() {
    List<Actividad> temp = new ArrayList();
    for (Actividad a : super.getList()) {
        Actividad m = (Actividad) a;
        temp.add(m);
    }

    if (temp.size() > 0 && temp != null) {
        return temp.get(temp.size() - 1).getCodigo() + 1;
    } else {
        return 1;
    }
}

public void ingresarRegex(String regex) {
    patron = Pattern.compile(regex);
}

public Set<String> obtenerUrl(String texto) {

    Set<String> resultado = new HashSet();
    corpus = patron.matcher(texto);

    while (corpus.find()) {
        resultado.add(corpus.group(0));
    }
    return resultado;
}

public List<Actividad> actividades() {

    List<Actividad> lista = new ArrayList();
    Actividad actividad;
    Iterator i = super.getList().iterator();
    while (i.hasNext()) {
        actividad = (Actividad) i.next();
        lista.add(actividad);
    }
}

```



```
}  
    return lista;  
  
}  
  
}  
  
➤ Controlador Alumno  
package ec.edu.ups.controlador;  
  
import ec.edu.ups.modelo.Alumno;  
import java.util.ArrayList;  
import java.util.Iterator;  
import java.util.List;  
  
/**  
 *  
 * @author paul_  
 */  
public class ControladorAlumno extends AbstractControlador<Alumno> {  
  
    private ControladorPersona control;  
  
    public ControladorAlumno(ControladorPersona control, String ruta) {  
        super(ruta);  
        this.control = control;  
    }  
  
    @Override  
    public boolean validar(Alumno objeto) {  
        return control.validar(objeto);  
    }  
  
    @Override  
    public int generarId() {  
        return 0;  
    }  
  
    public List<Alumno> alumnos() {  
  
        List<Alumno> lista = new ArrayList();  
        Alumno alumno;  
        Iterator i = super.getList().iterator();  
        while (i.hasNext()) {  
            alumno = (Alumno) i.next();  
            lista.add(alumno);  
        }  
        return lista;  
    }  
  
}
```

➤ Controlador Curso

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

import ec.edu.ups.modelo.Curso;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

/**
 *
 * @author paul_
 */
public class ControladorCurso extends AbstractControlador<Curso> {

    public ControladorCurso(String ruta) {
        super(ruta);
    }

    @Override
    public boolean validar(Curso objeto) {
        return true;
    }

    @Override
    public int generarId() {
        List<Curso> temp = new ArrayList();
        for (Curso a : super.getLista()) {
            Curso m = (Curso) a;
            temp.add(m);
        }

        if (temp.size() > 0 && temp != null) {
            return temp.get(temp.size() - 1).getCodigo() + 1;
        } else {
            return 1;
        }
    }

    public List<Curso> cursos() {

        List<Curso> lista = new ArrayList();
        Curso curso;
        Iterator i = super.getLista().iterator();
        while (i.hasNext()) {
            curso = (Curso) i.next();
            lista.add(curso);
        }
        return lista;
    }
}
```

```
}

➤ Controlador Persona
package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Persona;

/**
 *
 * @author paul_
 */
public class ControladorPersona extends AbstractControlador<Persona> {

    public ControladorPersona(String ruta) {
        super(ruta);
    }

    @Override
    public boolean validar(Persona objeto) {
        int suma = 0;
        String x = objeto.getCedula();
        if (x.length() == 9) {
            return false;
        } else {
            int a[] = new int[x.length() / 2];
            int b[] = new int[(x.length() / 2)];
            int c = 0;
            int d = 1;
            for (int i = 0; i < x.length() / 2; i++) {
                a[i] = Integer.parseInt(String.valueOf(x.charAt(c)));
                c = c + 2;
                if (i < (x.length() / 2) - 1) {
                    b[i] = Integer.parseInt(String.valueOf(x.charAt(d)));
                    d = d + 2;
                }
            }

            for (int i = 0; i < a.length; i++) {
                a[i] = a[i] * 2;
                if (a[i] > 9) {
                    a[i] = a[i] - 9;
                }
                suma = suma + a[i] + b[i];
            }
            int aux = suma / 10;
            int dec = (aux + 1) * 10;
            if ((dec - suma) == Integer.parseInt(String.valueOf(x.charAt(x.length()
- 1)))) {
                return true;
            } else if (suma % 10 == 0 && x.charAt(x.length() - 1) == '0') {
                return true;
            } else {
                return false;
            }
        }
    }
}
```

```
    }  
    }  
}  
  
@Override  
public int generarId() {  
    return 0;  
}  
  
}  
  
➤ Controlador Profesor  
package ec.edu.ups.controlador;  
  
import ec.edu.ups.modelo.Profesor;  
import java.util.ArrayList;  
import java.util.Iterator;  
import java.util.List;  
  
/**  
 *  
 * @author paul_  
 */  
public class ControladorProfesor extends AbstractControlador<Profesor> {  
  
    private ControladorPersona control;  
  
    public ControladorProfesor(ControladorPersona control, String ruta) {  
        super(ruta);  
        this.control = control;  
    }  
  
    @Override  
    public boolean validar(Profesor objeto) {  
        return control.validar(objeto);  
    }  
  
    @Override  
    public int generarId() {  
        return 0;  
    }  
  
    public List<Profesor> docentes() {  
  
        List<Profesor> lista = new ArrayList();  
        Profesor docente;  
        Iterator i = super.getList().iterator();  
        while (i.hasNext()) {  
            docente = (Profesor) i.next();  
            lista.add(docente);  
        }  
    }  
}
```

```
        return lista;
    }

}

➤ Controlador Rector
package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Profesor;
import ec.edu.ups.modelo.Rector;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

/**
 *
 * @author paul_
 */
public class ControladorRector extends AbstractControlador<Rector>{

    private Rector usuario;

    public ControladorRector(String ruta) {
        super(ruta);
    }


    @Override
    public boolean validar(Rector objeto) {
        return true;
    }

    @Override
    public int generarId() {
        return 0;
    }

    public List<Rector> usuarios() {

        List<Rector> lista = new ArrayList();
        Rector u;
        Iterator i = super.getList().iterator();
        while (i.hasNext()) {
            u = (Rector) i.next();
            lista.add(u);
        }
        return lista;
    }

    public Rector getUsuario() {
        return usuario;
    }
}
```

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

```

}

public boolean iniciarSesion(String correo, String pass) {

    for (Rector usu : super.getLista()) {
        Rector u = (Rector) usu;
        if (u.getCorreo().equals(correo) && u.getPass().equals(pass)) {
            this.usuario = u;
            return true;
        }
    }
    return false;
}

public Rector buscarU(Rector docente) {

    for (Rector usu : super.getLista()) {
        Rector u = (Rector) usu;
        if (docente.equals(u.getProfesor())) {
            this.usuario = u;

        }
    }
    return null;
}

public Rector buscarU(Profesor doc) {
    throw new UnsupportedOperationException("Not supported yet."); //To change
body of generated methods, choose Tools | Templates.
}

}

-ec.edu.ups.modelo
➤ Actividad
package ec.edu.ups.modelo;

import java.io.Serializable;

/**
 *
 * @author paul_
 */
public class Actividad implements Serializable {

    private int codigo;
    private String titulo;
    private String descripcion;
    private String link;
    private Curso curso;

    public Actividad(int codigo, String titulo, String descripcion, String link,
Curso curso) {
        this.codigo = codigo;

```

```
this.titulo = titulo;
this.descripcion = descripcion;
this.link = link;
this.curso = curso;
}

public int getCodigo() {
    return codigo;
}

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

public String getTitulo() {
    return titulo;
}

public void setTitulo(String titulo) {
    this.titulo = titulo;
}

public String getDescripcion() {
    return descripcion;
}

public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
}

public String getLink() {
    return link;
}


public void setLink(String link) {
    this.link = link;
}

public Curso getCurso() {
    return curso;
}

public void setCurso(Curso curso) {
    this.curso = curso;
}

@Override
public int hashCode() {
    int hash = 7;
    hash = 71 * hash + this.codigo;
    return hash;
}

@Override
```

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

```

public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null) {
        return false;
    }
    if (getClass() != obj.getClass()) {
        return false;
    }
    final Actividad other = (Actividad) obj;
    if (this.codigo != other.codigo) {
        return false;
    }
    return true;
}

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append("Actividad{codigo=").append(codigo);
    sb.append(", titulo=").append(titulo);
    sb.append(", descripcion=").append(descripcion);
    sb.append(", link=").append(link);
    sb.append(", curso=").append(curso);
    sb.append('}');
    return sb.toString();
}

}

➤ Alumno
package ec.edu.ups.modelo;

/**
 *
 * @author paul_
 */
public class Alumno extends Persona{


    private String aula;
    private Curso curso;

    public Alumno(String cedula, String nombre, String apellido, int edad, String
direccion, Curso curso, String aula) {
        super(cedula, nombre, apellido, edad, direccion);
        this.curso = curso;
        this.aula = aula;
    }

    public String getAula() {
        return aula;
    }

    public void setAula(String aula) {

```


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

```

        this.aula = aula;
    }

    public Curso getCurso() {
        return curso;
    }

    public void setCurso(Curso curso) {
        this.curso = curso;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("Alumno{aula=").append(aula);
        sb.append(", curso=").append(curso);
        sb.append('}');
        return sb.toString();
    }
}

➤ Curso
package ec.edu.ups.modelo;

import java.io.Serializable;

/**
 *
 * @author paul_
 */
public class Curso implements Serializable{

    private int codigo;
    private String nombre;
    private String seccion;

    public Curso(int codigo, String nombre, String seccion) {
        this.codigo = codigo;
        this.nombre = nombre;
        this.seccion = seccion;
    }

    public Curso() {
        throw new UnsupportedOperationException("Not supported yet."); //To change
body of generated methods, choose Tools | Templates.
    }

    public int getCodigo() {
        return codigo;
    }

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

```

```
public String getNombre() {
    return nombre;
}

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


public String getSeccion() {
    return seccion;
}

public void setSeccion(String seccion) {
    this.seccion = seccion;
}

@Override
public int hashCode() {
    int hash = 3;
    hash = 29 * hash + this.codigo;
    return hash;
}

@Override
public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null) {
        return false;
    }
    if (getClass() != obj.getClass()) {
        return false;
    }
    final Curso other = (Curso) obj;
    if (this.codigo != other.codigo) {
        return false;
    }
    return true;
}

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append("Curso{codigo=").append(codigo);
    sb.append(", nombre=").append(nombre);
    sb.append(", seccion=").append(seccion);
    sb.append('}');
    return sb.toString();
}
}
```

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

➤ Persona

```
package ec.edu.ups.modelo;

import java.io.Serializable;
import java.util.Objects;

/**
 *
 * @author paul_
 */
public class Persona implements Serializable{

    private String cedula;
    private String nombre;
    private String apellido;
    private int edad;
    private String direccion;

    public Persona(String cedula, String nombre, String apellido, int edad, String
direccion) {
        this.cedula = cedula;
        this.nombre = nombre;
        this.apellido = apellido;
        this.edad = edad;
        this.direccion = direccion;
    }

    public String getCedula() {
        return cedula;
    }

    public void setCedula(String cedula) {
        this.cedula = cedula;
    }


    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;
    }

    public int getEdad() {
        return edad;
    }
}
```

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

```

public void setEdad(int edad) {
    this.edad = edad;
}

public String getDireccion() {
    return direccion;
}

public void setDireccion(String direccion) {
    this.direccion = direccion;
}

@Override
public int hashCode() {
    int hash = 3;
    hash = 37 * hash + Objects.hashCode(this.cedula);
    return hash;
}


@Override
public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null) {
        return false;
    }
    if (getClass() != obj.getClass()) {
        return false;
    }
    final Persona other = (Persona) obj;
    if (!Objects.equals(this.cedula, other.cedula)) {
        return false;
    }
    return true;
}

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append("Persona{cedula=").append(cedula);
    sb.append(", nombre=").append(nombre);
    sb.append(", apellido=").append(apellido);
    sb.append(", edad=").append(edad);
    sb.append(", direccion=").append(direccion);
    sb.append('}');
    return sb.toString();
}
}

```

➤ Profesor

```
package ec.edu.ups.modelo;
```

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

```

/**
 *
 * @author paul_
 */
public class Profesor extends Persona{

    private String titulo;
    private String tipoTitulo;
    private Curso curso;
    private Rector rector;

    public Profesor(String cedula, String nombre, String apellido, int edad, String
direccion) {
        super(cedula, nombre, apellido, edad, direccion);
    }

    public Profesor(String cedula, String nombre, String apellido, int edad, String
direccion, String titulo, String tipoTitulo) {
        super(cedula, nombre, apellido, edad, direccion);
        this.tipoTitulo = tipoTitulo;
        this.titulo = titulo;
    }

    public String getTitulo() {
        return titulo;
    }

    public void setTitulo(String titulo) {
        this.titulo = titulo;
    }

    public String getTipoTitulo() {
        return tipoTitulo;
    }

    public void setTipoTitulo(String tipoTitulo) {
        this.tipoTitulo = tipoTitulo;
    }


    public Curso getCurso() {
        return curso;
    }

    public void setCurso(Curso curso) {
        this.curso = curso;
    }

    public Rector getRector() {
        return rector;
    }

    public void setRector(Rector rector) {
        this.rector = rector;
    }
}

```

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

```

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append("Profesor{titulo=").append(titulo);
    sb.append(", tipoTitulo=").append(tipoTitulo);
    sb.append(", curso=").append(curso);
    sb.append(", rector=").append(reactor);
    sb.append('}');
    return sb.toString();
}

}
➤ Rector
package ec.edu.upi.modelo;

import java.io.Serializable;

/**
 *
 * @author paul_
 */
public class Rector implements Serializable{

    private Profesor profesor;
    private String rol;
    private String correo;
    private String pass;

    public Rector(Profesor profesor, String rol, String correo, String pass) {
        this.profesor = profesor;
        this.rol = rol;
        this.correo = correo;
        this.pass = pass;
    }

    public Profesor getProfesor() {
        return profesor;
    }

    public void setProfesor(Profesor profesor) {
        this.profesor = profesor;
    }

    public String getRol() {
        return rol;
    }

    public void setRol(String rol) {
        this.rol = rol;
    }

    public String getCorreo() {
        return correo;
    }

```

```
}

public void setCorreo(String correo) {
    this.correo = correo;
}

public String getPass() {
    return pass;
}

public void setPass(String pass) {
    this.pass = pass;
}

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append("Rector{profesor=").append(profesor);
    sb.append(", rol=").append(rol);
    sb.append(", correo=").append(correo);
    sb.append(", pass=").append(pass);
    sb.append('}');
    return sb.toString();
}
}
```

-ec.edu.ups.vista

➤ Gestión Actividades

```
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorActividad;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import java.util.Set;


/**
 *
 * @author paul_
 */
public class GestionActividades extends javax.swing.JInternalFrame {

    private ControladorActividad controladorActividad;

    private String textoBusqueda;
    private String ruta = "datos/Actividad.obj";

    public GestionActividades(ControladorActividad controladorActividad) {
        initComponents();
        this.controladorActividad = controladorActividad;

        textoBusqueda = txtActividad.getText().trim();
    }
}
```

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

```

    }
private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose();
}

private void btnLimpiarActionPerformed(java.awt.event.ActionEvent evt) {
    txtActividad.setText("");
    Areatxt.setText("");
}

private void btnIniciarActionPerformed(java.awt.event.ActionEvent evt) {
    controladorActividad.ingresarRegex("<a\\shref=\\\"\\S+\"");
    StringBuilder sb = new StringBuilder();
    StringBuilder mostrar = new StringBuilder();//&c=apps
    try {
        URL urlObject = new
URL("https://play.google.com/store/search?q="+textoBusqueda+"&c=apps");
        URLConnection urlC = urlObject.openConnection();
        urlC.setRequestProperty("User-Agent", "Mozilla/5.0 (Windows NT 6.1;
WOW64) AppleWebKit/537.11 (KHTML, like Gecko) Chrome/23.0.1271.95 Safari/537.11");
        BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(urlC.getInputStream(), "UTF-8"));
        String entrada;
        while ((entrada = bufferedReader.readLine()) != null) {
            sb.append(entrada);
        }
    } catch (IOException e) {

    }

    Set<String> res = controladorActividad.obtenerUrl(sb.toString());//
res.stream().forEach(s ->mostrar.append(s.replaceAll("<a\\shref=\\\"",
""))+ "\\n"));
    Areatxt.setText(mostrar.toString());

}

// Variables declaration - do not modify
private javax.swing.JTextArea Areatxt;
private javax.swing.JButton btnCancelar;
private javax.swing.JButton btnIniciar;
private javax.swing.JButton btnLimpiar;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel5;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTextField txtActividad;
// End of variables declaration
}

```


ACTIVIDADES EN LAS QUE SE PUEDE TRABAJAR:

Ingrese Actividad

BUSCAR

LIMPIAR

SALIR

➤ Gestión Cursos

```
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorCurso;
import ec.edu.ups.modelo.Curso;
import java.io.IOException;
import java.util.Optional;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author paul_
 */
public class GestionCursos extends javax.swing.JInternalFrame {

    private ControladorCurso controladorCurso;
    public static String ruta = "datos/Curso.obj";
```

```
public GestionCursos(ControladorCurso controladorCurso) {

    initComponents();

    this.controladorCurso = controladorCurso;

}

public void cargarSiguienteCodigo() {
    txtCodigo.setText("" + controladorCurso.generarId());
}

public void cargarCursosTbl() {
    DefaultTableModel modelo = (DefaultTableModel) tblCursos.getModel();
    modelo.setRowCount(0);
    if (controladorCurso.cursos() != null) {
        for (Curso curso : controladorCurso.cursos()) {
            Object[] rowData = {curso.getCodigo(), curso.getNombre(),
curso.getSeccion()};
            modelo.addRow(rowData);
            tblCursos.setModel(modelo);
        }
    } else {
        System.out.println("LISTA VACIA");
    }
}

private void btnListarActionPerformed(java.awt.event.ActionEvent evt) {
    if (controladorCurso.cursos() != null) {


        cargarCursosTbl();

    }
}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    int d = JOptionPane.showConfirmDialog(this, "Esta seguro de eliminar el
curso con codigo" + txtCodigo.getText().trim());
    if (d == JOptionPane.YES_OPTION) {
        Curso comparar = new Curso(Integer.parseInt(txtCodigo.getText().trim()),
null, null);

        Optional<Curso> p = controladorCurso.buscar(comparar);
        Curso doc = p.get();
        System.out.println("" + doc);

        if (controladorCurso.eliminar(doc)) {
            JOptionPane.showMessageDialog(this, "eliminado exitosamente");
            try {
                controladorCurso.guardarDatos(ruta);
            } catch (IOException ex) {
```

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

```

Logger.getLogger(GestionCursos.class.getName()).log(Level.SEVERE, null, ex);
    }
    limpiar();
    cargarCursosTbl();
} else if (d == JOptionPane.NO_OPTION) {
    cargarCursosTbl();
}

}

}

private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {

    Curso curso = new Curso(Integer.parseInt(txtCodigo.getText().trim()),
txtnombre.getText().trim(), txtSeccion.getText().trim());

    if (controladorCurso.actualizar(curso)) {
        JOptionPane.showMessageDialog(this, "actualizado con exito");
    if (controladorCurso.crear(curso)) {
        JOptionPane.showMessageDialog(this, "Registrado con exito");
        try {
            controladorCurso.guardarDatos(ruta);
        } catch (IOException ex) {

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

        cargarCursosTbl();
        limpiar();

    } else {
        JOptionPane.showMessageDialog(this, "No se pudo actualizar");
    }
}

}


private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {

    Curso curso = new Curso(Integer.parseInt(txtCodigo.getText().trim()),
txtnombre.getText().trim(), txtSeccion.getText().trim());

    if (controladorCurso.crear(curso)) {
        JOptionPane.showMessageDialog(this, "Registrado con exito");
        try {
            controladorCurso.guardarDatos(ruta);
        } catch (IOException ex) {
            Logger.getLogger(GestionCursos.class.getName()).log(Level.SEVERE,
null, ex);
        }
        limpiar();
        cargarCursosTbl();

    } else {

```

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

```

        JOptionPane.showMessageDialog(this, "No se pudo registrar");
    }

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
    }

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

    private void tblCursosMouseClicked(java.awt.event.MouseEvent evt) {
        int index = tblCursos.getSelectedRow();

        String codigo = "" + tblCursos.getValueAt(index, 0);
        String nombre = "" + tblCursos.getValueAt(index, 1);
        String seccion = "" + tblCursos.getValueAt(index, 2);


        txtCodigo.setText(codigo);
        txtnombre.setText(nombre);
        txtSeccion.setText(seccion);

    }

    private void formInternalFrameActivated(javax.swing.event.InternalFrameEvent
evt) {
        cargarSiguienteCodigo();
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnActualizar;
    private javax.swing.JButton btnCancelar;
    private javax.swing.JButton btnEliminar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JButton btnListar;
    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.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable tblCursos;
    private javax.swing.JTextField txtCodigo;
    private javax.swing.JTextField txtSeccion;
    private javax.swing.JTextField txtnombre;
    // 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		

GESTION DE CURSOS:

Codigo

Nombre:

Seccion:

AGREGAR

MODIFICAR

ELIMINAR

LISTAR

SALIR

LISTA DE CURSOS:

Codigo	Nombre	Seccion


➤ Gestión Docentes

```
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorCurso;
import ec.edu.ups.controlador.ControladorProfesor;
import ec.edu.ups.controlador.ControladorRector;
import ec.edu.ups.modelo.Curso;
import ec.edu.ups.modelo.Profesor;
import ec.edu.ups.modelo.Rector;
import java.io.IOException;
import java.util.Optional;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author paul_
 */
public class GestionDocentes extends javax.swing.JFrame {

    private ControladorProfesor controladorDocente;
    private ControladorCurso controladorCurso;
    private ControladorRector controladorUsuario;
    public static String ruta = "datos/Docente.obj";
```

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

```

public GestionDocentes(ControladorProfesor controladorDocente, ControladorCurso
controladorCurso, ControladorRector controladorUsuario) {
    initComponents();

    this.controladorDocente = controladorDocente;
    this.controladorCurso = controladorCurso;
    this.controladorUsuario = controladorUsuario;
    try {
        controladorDocente.cargarDatos();
    } catch (ClassNotFoundException | IOException ex) {
        Logger.getLogger(GestionDocentes.class.getName()).log(Level.SEVERE,
null, ex);
    }

}

public void cargarDocentesTbl() {
    DefaultTableModel modelo = (DefaultTableModel) tblDocentes.getModel();
    modelo.setRowCount(0);
    if (controladorDocente.docentes() != null) {
        for (Profesor docente : controladorDocente.docentes()) {
            Object[] rowData = {docente.getCedula(), docente.getNombre(),
docente.getApellido(),
docente.getEdad(), docente.getDireccion(), docente.getTitulo(),
docente.getTipoTitulo(), docente.getCurso().getCodigo()};
            modelo.addRow(rowData);
            tblDocentes.setModel(modelo);
        }
    } else {
        System.out.println("LISTA VACIA");
    }

}

public void limpiar() {

    txtGcedula.setText("");
    txtGnombre.setText("");
    txtGapellido.setText("");
    txtGedad.setText("");
    txtGDireccion.setText("");
    txtGtitulo.setText("");
    txtGTipoTitulo.setText("");


}

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

private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose();
}

private void btnGuardarActionPerformed(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		

```

    Profesor docente = new Profesor(txtGcedula.getText().trim(),
txtGnombre.getText().trim(),
        txtGapellido.getText().trim(),
Integer.parseInt(txtGEdad.getText().trim()), txtGDireccion.getText().trim(),
txtTitulo.getText().trim(), txtTipoTitulo.getText().trim());

    Curso comparar = new Curso();
    comparar.setCodigo(Integer.parseInt(txtCodigoCurso.getText().trim()));

    Optional<Curso> c = controladorCurso.buscar(comparar);
    Curso curso = c.get();
    docente.setCurso(curso);
    Rector u = new Rector(docente, "editor", txtCorreo.getText().trim(),
txtPass.getText().trim());

    if (controladorDocente.crear(docente) && controladorUsuario.crear(u)) {
        cargarDocentesTbl();
        JOptionPane.showMessageDialog(this, "Registrado con exito");
        try {
            controladorDocente.guardarDatos(ruta);
            controladorUsuario.guardarDatos("datos/Usuario.obj");
        } catch (IOException ex) {
            Logger.getLogger(GestionDocentes.class.getName()).log(Level.SEVERE,
null, ex);
        }
        cargarDocentesTbl();
        limpiar();
    } else {
        JOptionPane.showMessageDialog(this, "No se pudo registrar");
    }
}

private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {

    Profesor docente = new Profesor(txtGcedula.getText().trim(),
txtGnombre.getText().trim(),
        txtGapellido.getText().trim(),
Integer.parseInt(txtGEdad.getText().trim()), txtGDireccion.getText().trim(),
txtTitulo.getText().trim(), txtTipoTitulo.getText().trim());

    Curso comparar = new Curso();
    comparar.setCodigo(Integer.parseInt(txtCodigoCurso.getText().trim()));

    Optional<Curso> c = controladorCurso.buscar(comparar);
    Curso curso = c.get();
    docente.setCurso(curso);
    Rector u = new Rector(docente, "editor", txtCorreo.getText().trim(),
txtPass.getText().trim());

    if (controladorDocente.actualizar(docente) &&
controladorUsuario.actualizar(u)) {
        JOptionPane.showMessageDialog(this, "actualizado exitosamente");
    }
}

```


```
        try {
            controladorDocente.guardarDatos(ruta);
            controladorUsuario.guardarDatos("datos/Usuario.obj");
        } catch (IOException ex) {
            Logger.getLogger(GestionDocentes.class.getName()).log(Level.SEVERE,
null, ex);
        }
        limpiar();
        cargarDocentesTbl();
    } else {
        JOptionPane.showMessageDialog(this, "No se pudo actualizar :ERROR");
    }
}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    int d = JOptionPane.showConfirmDialog(this, "Esta seguro de eliminar el
docente con cedula" + txtGcedula.getText().trim());
    if (d == JOptionPane.YES_OPTION) {
        Profesor comparar = new Profesor(txtGcedula.getText().trim(), null,
null, 0, null);
        comparar.setCedula(txtGcedula.getText().trim());
        Optional<Profesor> p = controladorDocente.buscar(comparar);
        Profesor doc = p.get();
        System.out.println("" + doc);
        Rector compa = controladorUsuario.buscarU(doc);

        if (controladorDocente.eliminar(doc) &&
controladorUsuario.eliminar(compa)) {
            try {
                controladorDocente.guardarDatos(ruta);
                controladorUsuario.guardarDatos("datos/Usuario.obj");
            } catch (IOException ex) {
                Logger.getLogger(GestionDocentes.class.getName()).log(Level.SEVERE, null, ex);
            }
            JOptionPane.showMessageDialog(this, "eliminado exitosamente");
            cargarDocentesTbl();
        } else if (d == JOptionPane.NO_OPTION) {
            cargarDocentesTbl();
        }
    }
}

private void btnListarActionPerformed(java.awt.event.ActionEvent evt) {
    if (controladorDocente.docentes() != null) {

        cargarDocentesTbl();
    }
}
```


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

```


private void tblDocentesMouseClicked(java.awt.event.MouseEvent evt) {
    int index = tblDocentes.getSelectedRow();

    String cedula = "" + tblDocentes.getValueAt(index, 0);
    String nombre = "" + tblDocentes.getValueAt(index, 1);
    String apellido = "" + tblDocentes.getValueAt(index, 2);
    String Edad = "" + tblDocentes.getValueAt(index, 3);
    String Direccion = "" + tblDocentes.getValueAt(index, 4);
    String Titulo = "" + tblDocentes.getValueAt(index, 5);
    String tipoTitulo = "" + tblDocentes.getValueAt(index, 6);
    int codigoCurso = Integer.parseInt("" + tblDocentes.getValueAt(index, 7));
    String correo = "" + tblDocentes.getValueAt(index, 8);
    String pass = "" + tblDocentes.getValueAt(index, 9);

    txtGcedula.setText(cedula);
    txtGnombre.setText(nombre);
    txtGapellido.setText(apellido);
    txtGEdad.setText(Edad);
    txtGDireccion.setText(Direccion);
    txtGTitulo.setText(Titulo);
    txtGTipoTitulo.setText(tipoTitulo);
    txtCodigoCurso.setText("" + codigoCurso);
    txtCorreo.setText(correo);
    txtPass.setText(pass);
}

// Variables declaration - do not modify
private javax.swing.JButton btnActualizar;
private javax.swing.JButton btnCancelar;
private javax.swing.JButton btnEliminar;
private javax.swing.JButton btnGuardar;
private javax.swing.JButton btnListar;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
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.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JLabel jLabel9;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTable tblDocentes;
private javax.swing.JTextField txtCodigoCurso;
private javax.swing.JTextField txtCorreo;
private javax.swing.JTextField txtGDireccion;
private javax.swing.JTextField txtGEdad;
private javax.swing.JTextField txtGapellido;
private javax.swing.JTextField txtGcedula;
private javax.swing.JTextField txtGnombre;

```

	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.JTextField txtPass;
private javax.swing.JTextField txtTipoTitulo;
private javax.swing.JTextField txtTitulo;
// End of variables declaration
}
```

GESTION DE DOCENTES:

Cedula:

Nombre:

Apellido:

Edad:

Direccion:

Título:

Tipo Título:

Codigo de Curso:

Correo:

Correo:


Cedula	Nombre	Apellido	Edad	Direccion	Título	Tipo título	codigo de curso

➤ Gestión Estudiantes

```
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorAlumno;
import ec.edu.ups.controlador.ControladorCurso;
import ec.edu.ups.modelo.Alumno;
import ec.edu.ups.modelo.Curso;
import java.io.IOException;
import java.util.Optional;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author paul_
```

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

```

*/
public class GestionEstudiantes extends javax.swing.JInternalFrame {

    public static String ruta = "datos/Alumno.obj";
    private ControladorAlumno controladorAlumno;
    private ControladorCurso controladorCurso;

    public GestionEstudiantes(ControladorAlumno controladorAlumno, ControladorCurso
controladorCurso) {
        initComponents();

        this.controladorAlumno = controladorAlumno;
        this.controladorCurso = controladorCurso;
    }

    public void cargarEstudiantesTbl() {
        DefaultTableModel modelo = (DefaultTableModel) tblAlumnos.getModel();
        modelo.setRowCount(0);
        if (controladorAlumno.alumnos() != null) {
            for (Alumno alumno : controladorAlumno.alumnos()) {
                Object[] rowData = {alumno.getCedula(), alumno.getNombre(),
alumno.getApellido(), alumno.getEdad(),
alumno.getDireccion(), alumno.getCurso().getCodigo(),
alumno.getAula()};
                modelo.addRow(rowData);
                tblAlumnos.setModel(modelo);
            }
        } else {
            System.out.println("LISTA VACIA");
        }
    }

    public void limpiar() {

        txtGcedula.setText("");
        txtGnombre.setText("");
        txtGapellido.setText("");
        txtGedad.setText("");
        txtGDireccion.setText("");
        txtCodigoCurso.setText("");
        txtAula.setText("");


    }

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

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        Curso comparar = new Curso();
        comparar.setCodigo(Integer.parseInt(txtCodigoCurso.getText().trim()));
    }
}

```

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

```

Optional<Curso> c = controladorCurso.buscar(comparar);
Curso curso = c.get();
Alumno estudiante = new Alumno(txtGcedula.getText().trim(),
txtGnombre.getText().trim(), txtGapellido.getText().trim(),
Integer.parseInt(txtGEdad.getText().trim()),
txtGDireccion.getText().trim(), curso, txtAula.getText().trim());

if (controladorAlumno.crear(estudiante)) {
    try {
        JOptionPane.showMessageDialog(this, "Registrado con exito");

        controladorAlumno.guardarDatos(ruta);
        cargarEstudiantesTbl();
        limpiar();
    } catch (IOException ex) {

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

    } else {
        JOptionPane.showMessageDialog(this, "No se pudo registrar");
    }
}

private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {

Curso comparar = new Curso();
comparar.setCodigo(Integer.parseInt(txtCodigoCurso.getText().trim()));


Optional<Curso> c = controladorCurso.buscar(comparar);
Curso curso = c.get();
Alumno estudiante = new Alumno(txtGcedula.getText().trim(),
txtGnombre.getText().trim(), txtGapellido.getText().trim(),
Integer.parseInt(txtGEdad.getText().trim()),
txtGDireccion.getText().trim(), curso, txtAula.getText().trim());
if (controladorAlumno.actualizar(estudiante)) {
    JOptionPane.showMessageDialog(this, "actualizado exitosamente");

    try {
        controladorAlumno.guardarDatos(ruta);
    } catch (IOException ex) {

Logger.getLogger(GestionEstudiantes.class.getName()).log(Level.SEVERE, null, ex);
    }
    cargarEstudiantesTbl();
    limpiar();

    limpiar();
    cargarEstudiantesTbl();
} else {
    JOptionPane.showMessageDialog(this, "No se pudo actualizar :ERROR");
}
}

```

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

```

}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    int d = JOptionPane.showConfirmDialog(this, "Esta seguro de eliminar el
docente con cedula" + txtGcedula.getText().trim());
    if (d == JOptionPane.YES_OPTION) {
        Alumno comparar = new Alumno(txtGcedula.getText().trim(), null, null, 0,
null, null, null);
        comparar.setCedula(txtGcedula.getText().trim());
        Optional<Alumno> p = controladorAlumno.buscar(comparar);
        Alumno doc = p.get();
        System.out.println("" + doc);

        if (controladorAlumno.eliminar(doc)) {
            JOptionPane.showMessageDialog(this, "eliminado exitosamente");
            try {
                controladorAlumno.guardarDatos(ruta);
            } catch (IOException ex) {

                Logger.getLogger(GestionEstudiantes.class.getName()).log(Level.SEVERE, null, ex);
            }
            cargarEstudiantesTbl();
        } else if (d == JOptionPane.NO_OPTION) {
            cargarEstudiantesTbl();
        }

    }

}

private void btnListarActionPerformed(java.awt.event.ActionEvent evt) {
    if (controladorAlumno.alumnos() != null) {

        cargarEstudiantesTbl();

    }


}

private void tblAlumnosMouseClicked(java.awt.event.MouseEvent evt) {
    int index = tblAlumnos.getSelectedRow();

    String cedula = "" + tblAlumnos.getValueAt(index, 0);
    String nombre = "" + tblAlumnos.getValueAt(index, 1);
    String apellido = "" + tblAlumnos.getValueAt(index, 2);
    String Edad = "" + tblAlumnos.getValueAt(index, 3);
    String Direccion = "" + tblAlumnos.getValueAt(index, 4);
    int codigoCurso = Integer.parseInt("" + tblAlumnos.getValueAt(index, 5));
    String aula = "" + tblAlumnos.getValueAt(index, 6);

    txtGcedula.setText(cedula);
    txtGnombre.setText(nombre);
    txtGapellido.setText(apellido);
    txtGEdad.setText(Edad);

```

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


```

txtGDireccion.setText(Direccion);
txtCodigoCurso.setText("" + codigoCurso);
txtAula.setText(aula);

}

// Variables declaration - do not modify
private javax.swing.JButton btnActualizar;
private javax.swing.JButton btnCancelar;
private javax.swing.JButton btnEliminar;
private javax.swing.JButton btnGuardar;
private javax.swing.JButton btnListar;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
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.JLabel jLabel9;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTable tblAlumnos;
private javax.swing.JTextField txtAula;
private javax.swing.JTextField txtCodigoCurso;
private javax.swing.JTextField txtGDireccion;
private javax.swing.JTextField txtGEdad;
private javax.swing.JTextField txtGapellido;
private javax.swing.JTextField txtGcedula;
private javax.swing.JTextField txtGnombre;
// 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		

GESTION DE ESTUDIANTES

Cedula:

Nombre:

Apellido:

Edad:

Direccion:

Codigo de Curso:

Aula:

Cedula	Nombre	Apellido	Edad	Direccion	codigo de Curso	Aula


➤ Ventana Iniciar Sesión

```
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorRector;
import ec.edu.ups.modeloCurso;
import ec.edu.ups.modelo.Profesor;
import ec.edu.ups.modelo.Rector;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

/**
 *
 * @author paul_
 */
public final class VentanaIniciarSesion extends javax.swing.JInternalFrame {

    private VentanaPrincipal ventanaPrincipal;
    private ControladorRector controladorUsuario;
    private String ruta = "datos/Usuario.obj";
```

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

```

/**
 * Creates new form VentanaIniciarSesion
 *
 * @param ventanaPrincipal
 * @param controladorUsuario
 */
public VentanaIniciarSesion(VentanaPrincipal ventanaPrincipal, ControladorRector
controladorUsuario) throws IOException {
    initComponents();


    this.ventanaPrincipal = ventanaPrincipal;
    this.controladorUsuario = controladorUsuario;
    // generarAdmin();
    try {
        controladorUsuario.cargarDatos();
    } catch (ClassNotFoundException ex) {
        Logger.getLogger(VentanaIniciarSesion.class.getName()).log(Level.SEVERE,
null, ex);
    }
}

private void btnIniciarSesionActionPerformed(java.awt.event.ActionEvent evt) {
    //
    String pass = "";
    char[] pass1 = txtPass.getPassword();
    for (int i = 0; i < pass1.length; i++) {
        pass = pass + pass1[i];
    }

    if (controladorUsuario.iniciarSesion(txtCorreo.getText().trim(), pass)) {
        ventanaPrincipal.getIniciarSesionMenuItem().setVisible(false);
        ventanaPrincipal.getCerrarSesionMenuItem().setVisible(true);
        if (controladorUsuario.getUsuario().getRol().equals("admin")) {
            ventanaPrincipal.getCursoMenu().setVisible(true);
            ventanaPrincipal.getDocentesMenu().setVisible(true);
            ventanaPrincipal.getActividadesMenuItem().setVisible(false);
            ventanaPrincipal.getEstudiantesMenu().setVisible(false);
        } else {
            ventanaPrincipal.getCursoMenu().setVisible(false);
            ventanaPrincipal.getDocentesMenu().setVisible(false);
            ventanaPrincipal.getActividadesMenuItem().setVisible(true);
            ventanaPrincipal.getEstudiantesMenu().setVisible(true);
        }

        this.dispose();
        Limpiar();
        JOptionPane.showMessageDialog(this, "Inicio de sesion exitoso");
    } else {
        JOptionPane.showMessageDialog(this, "Usuario o contrasena incorrecta ");
        Limpiar();
    }
}

```


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

```

}
public void generarAdmin() {
    Curso curso = new Curso(1, "RECTORADO", "RECTOR GENERAL");
    Profesor docente = new Profesor("1303753618", "DIEGO", "QUISI", 48, "EL
VECINO", "MASTER EN SISTEMAS", "MASTERADO");
    docente.setCurso(curso);
    Rector rector = new Rector(docente, "admin", "admin", "admin");
    controladorUsuario.crear(rector);
    System.out.println("ADMIN GENERADO CORRECTAMENTE");


    try {
        controladorUsuario.guardarDatos(ruta);
    } catch (IOException ex) {
        Logger.getLogger(VentanaIniciarSesion.class.getName()).log(Level.SEVERE,
null, ex);
    }

}
private void formInternalFrameActivated(javax.swing.event.InternalFrameEvent
evt) {

}
public void Limpiar() {
    txtCorreo.setText("");
    txtPass.setText("");
}

// Variables declaration - do not modify
private javax.swing.JButton btnIniciarSesion;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel lblContra;
private javax.swing.JLabel lblCorreo;
private javax.swing.JTextField txtCorreo;
private javax.swing.JPasswordField txtPass;
// 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 Principal

```
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorActividad;
import ec.edu.ups.controlador.ControladorAlumno;
import ec.edu.ups.controlador.ControladorCurso;
import ec.edu.ups.controlador.ControladorProfesor;
import ec.edu.ups.controlador.ControladorPersona;
import ec.edu.ups.controlador.ControladorRector;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JDesktopPane;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;

/**
 *
 * @author paul_
 */
public class VentanaPrincipal extends javax.swing.JFrame {

    private VentanaIniciarSesion ventanaIniciarSesion;
    private GestionEstudiantes gestionEstudiantes;
    private GestionDocentes gestionDocentes;
    private GestionCursos gestionCursos;
    private GestionActividades gestionActividades;
    private ControladorPersona controladorPersona;
    private ControladorActividad controladorActividad;
    private ControladorAlumno controladorAlumno;
    private ControladorCurso controladorCurso;
    private ControladorProfesor controladorDocente;
    private ControladorRector controladorUsuario;
```

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

```

public VentanaPrincipal() {
    try {
        initComponents();

        controladorPersona = new ControladorPersona("datos/Persona.obj");
        controladorActividad = new ControladorActividad("datos/Actividad.obj");
        controladorAlumno = new ControladorAlumno(controladorPersona,
"datos/Alumno.obj");
        controladorCurso = new ControladorCurso("datos/Curso.obj");
        controladorDocente = new ControladorProfesor(controladorPersona,
"datos/Docente.obj");
        controladorUsuario = new ControladorRector("datos/Usuario.obj");

        ventanaIniciarSesion = new VentanaIniciarSesion(this,
controladorUsuario);
        gestionEstudiantes = new GestionEstudiantes(controladorAlumno,
controladorCurso);
        gestionDocentes = new GestionDocentes(controladorDocente,
controladorCurso, controladorUsuario);
        gestionCursos = new GestionCursos(controladorCurso);
        gestionActividades = new GestionActividades(controladorActividad);

        EstudiantesMenu.setVisible(false);
        DocentesMenu.setVisible(false);
        CursoMenu.setVisible(false);
        CerrarSesionMenuItem.setVisible(false);
    } catch (IOException ex) {
        Logger.getLogger(VentanaPrincipal.class.getName()).log(Level.SEVERE,
null, ex);
    }
}


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

private void CerrarSesionMenuItemActionPerformed(java.awt.event.ActionEvent evt)
{
    IniciarSesionMenuItem.setVisible(true);
    EstudiantesMenu.setVisible(false);
    DocentesMenu.setVisible(false);
    CursoMenu.setVisible(false);
    exitMenuItem.setVisible(true);
}

private void GestionEstMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    desktopPane.add(gestionEstudiantes);
    gestionEstudiantes.setVisible(true);
}

private void IniciarSesionMenuItemActionPerformed(java.awt.event.ActionEvent
evt) {
    desktopPane.add(ventanaIniciarSesion);
}

```

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

```

        ventanaIniciarSesion.setVisible(true);
    }

    private void formWindowClosing(java.awt.event.WindowEvent evt) {
    }

    private void GestionDocMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        desktopPane.add(gestionDocentes);
        gestionDocentes.setVisible(true);
    }

    private void ActividadesMenuItemActionPerformed(java.awt.event.ActionEvent evt)
    {
        desktopPane.add(gestionActividades);
        gestionActividades.setVisible(true);
    }

    private void GestionMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        desktopPane.add(gestionCursos);
        gestionCursos.setVisible(true);
    }

    public JMenuItem getCerrarSesionMenuItem() {
        return CerrarSesionMenuItem;
    }

    public void setCerrarSesionMenuItem(JMenuItem CerrarSesionMenuItem) {
        this.CerrarSesionMenuItem = CerrarSesionMenuItem;
    }

    public JMenu getCursoMenu() {
        return CursoMenu;
    }

    public void setCursoMenu(JMenu CursoMenu) {
        this.CursoMenu = CursoMenu;
    }


    public JMenu getDocentesMenu() {
        return DocentesMenu;
    }

    public void setDocentesMenu(JMenu DocentesMenu) {
        this.DocentesMenu = DocentesMenu;
    }

    public JMenu getEstudiantesMenu() {
        return EstudiantesMenu;
    }

    public void setEstudiantesMenu(JMenu EstudiantesMenu) {
        this.EstudiantesMenu = EstudiantesMenu;
    }

```

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

```

public JMenuItem getGestionDocMenuItem() {
    return GestionDocMenuItem;
}

public void setGestionDocMenuItem(JMenuItem GestionDocMenuItem) {
    this.GestionDocMenuItem = GestionDocMenuItem;
}

public JMenuItem getGestionEstMenuItem() {
    return GestionEstMenuItem;
}

public void setGestionEstMenuItem(JMenuItem GestionEstMenuItem) {
    this.GestionEstMenuItem = GestionEstMenuItem;
}

public JMenuItem getGestionMenuItem() {
    return GestionMenuItem;
}

public void setGestionMenuItem(JMenuItem GestionMenuItem) {
    this.GestionMenuItem = GestionMenuItem;
}

public JMenuItem getIniciarSesionMenuItem() {
    return IniciarSesionMenuItem;
}

public void setIniciarSesionMenuItem(JMenuItem IniciarSesionMenuItem) {
    this.IniciarSesionMenuItem = IniciarSesionMenuItem;
}

public JDesktopPane getDesktopPane() {
    return desktopPane;
}

public void setDesktopPane(JDesktopPane desktopPane) {
    this.desktopPane = desktopPane;
}


public JMenuItem getExitMenuItem() {
    return exitMenuItem;
}

public void setExitMenuItem(JMenuItem exitMenuItem) {
    this.exitMenuItem = exitMenuItem;
}

public JMenu getFileMenu() {
    return fileMenu;
}

public void setFileMenu(JMenu fileMenu) {

```

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

```

        this.fileMenu = fileMenu;
    }

    public JMenuItem getActividadesMenuItem() {
        return ActividadesMenuItem;
    }


    public void setActividadesMenuItem(JMenuItem ActividadesMenuItem) {
        this.ActividadesMenuItem = ActividadesMenuItem;
    }

    public void setMenuBar(JMenuBar menuBar) {
        this.menuBar = menuBar;
    }
// Variables declaration - do not modify
private javax.swing.JMenuItem ActividadesMenuItem;
private javax.swing.JMenuItem CerrarSesionMenuItem;
private javax.swing.JMenu CursoMenu;
private javax.swing.JMenu DocentesMenu;
private javax.swing.JMenu EstudiantesMenu;
private javax.swing.JMenuItem GestionDocMenuItem;
private javax.swing.JMenuItem GestionEstMenuItem;
private javax.swing.JMenuItem GestionMenuItem;
private javax.swing.JMenuItem IniciarSesionMenuItem;
private javax.swing.JDesktopPane desktopPane;
private javax.swing.JMenuItem exitMenuItem;
private javax.swing.JMenu fileMenu;
private javax.swing.JMenuBar menuBar;
// End of variables declaration

```

Inicio Estudiantes Docentes Cursos



	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):

- Interpreta de forma correcta los algoritmos de programación y su aplicabilidad.
- Identifica correctamente qué herramientas de programación se pueden aplicar.

CONCLUSIONES:

- Los estudiantes identifican las principales estructuras para la creación de sistemas informáticos.
- Los estudiantes implementan soluciones graficas en sistemas.
- Los estudiantes están en la capacidad de implementar la persistencia en archivos.

RECOMENDACIONES:

- Revisar la información proporcionada por el docente previo a la práctica.
- Haber asistido a las sesiones de clase.
- Consultar con el docente las dudas que puedan surgir al momento de realizar la prueba.

Nombre de estudiante:

PAUL ALEXANDER GUAPUCAL CARDENAS

Firma de estudiante:

