
	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 <p>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).</p> <p>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.</p> <p>Ejemplo Rector:</p> <p>Docentes:</p> <ol style="list-style-type: none"> 1. Diego Quisi 2. Vladimir Robles 3. Etc. <p>Cursos:</p> <ol style="list-style-type: none"> 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: Identificar los cambios importantes de Java Diseñar e Implementar expresiones regulares Entender la cada uno de las características nuevas en Java			
ACTIVIDADES DESARROLLADAS			
1. Codigo Desarrollado Paquete ec.edu.ups.modelo Clase Rector <pre> package ec.edu.ups.modelo; import java.io.Serializable; /** * * @author Anahi */ public class Rector implements Serializable{ private Docente docente; private String cargo; private String correo; private String contrasenia; public Rector(Docente docente, String cargo, String correo, String contrasenia) { this.docente = docente; this.cargo = cargo; this.correo = correo; this.contrasenia = contrasenia; } public String getCargo() { return cargo; } public void setCargo(String cargo) { this.cargo = cargo; } public String getContrasenia() { </pre>			

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

```

        return contrasenia;
    }

    public void setContrasenia(String contrasenia) {
        this.contrasenia = contrasenia;
    }

    public Docente getDocente() {
        return docente;
    }

    public void setDocente(Docente docente) {
        this.docente = docente;
    }

    public String getCorreo() {
        return correo;
    }

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

    @Override
    public String toString() {
        return "Rector{" + "docente=" + docente + ", cargo=" + cargo + ", correo=" +
correo + ", contrasenia=" + contrasenia + '}';
    }
}

```

Clase Alumno

```

package ec.edu.ups.modelo;

/**
 *
 * @author Anahi
 */
public class Alumno extends Persona{
    private Curso curso;

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

    public Curso getCurso() {

```

```
        return curso;
    }

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

    @Override
    public String toString() {
        return "Alumno{" + "curso=" + curso + '}';
    }
}

Clase Curso
package ec.edu.ups.modelo;

import java.util.List;

/**
 *
 * @author Anahi
 */
public class Curso {
    private int codigo;
    private String nivel;
    private String seccion;

    public Curso() {
    }

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

    public int getCodigo() {
        return codigo;
    }

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

    public String getNivel() {
        return nivel;
    }

    public void setNivel(String nivel) {
        this.nivel = nivel;
    }

    public String getSeccion() {
```

```
        return seccion;
    }

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

    @Override
    public int hashCode() {
        int hash = 7;
        hash = 67 * 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() {
        return "Curso{" + "codigo=" + codigo + ", nivel=" + nivel + ", seccion=" +
seccion + '}';
    }
}
```


Clase Actividad

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

import java.io.Serializable;

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

    private int codigo;
    private Curso curso;
```


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

```

private String link;
private String titulo;
private String descripcion;

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

public int getCodigo() {
    return codigo;
}

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

public Curso getCurso() {
    return curso;
}

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

public String getLink() {
    return link;
}

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

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;
}

@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 "Actividad{" + "codigo=" + codigo + ", curso=" + curso
        + ", link=" + link + ", titulo=" + titulo + ", descripcion=" +
descripcion + '}';
}

```

```

}

```

Clase Persona

```

package ec.edu.ups.modelo;

```

```

import java.io.Serializable;

```

```

import java.util.Objects;

```

```

/**

```

```

 *

```

```

 * @author Anahi

```

```

 */

```

```

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;
    }

    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 = 61 * 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() {
        return "Persona{" + "cedula=" + cedula + ", nombre=" + nombre +
            ", apellido=" + apellido + ", edad=" + edad + ", direccion="
            + direccion + '}';
    }
}
```

}

Clase Docente**public class Docente extends Persona{****private String profesion;**
private Curso curso;
private String correo;
private String contrasenia;**public Docente(String profesion, String correo, String contrasenia, String cedula, String nombre, String apellido, int edad, String direccion) {**
 super(cedula, nombre, apellido, edad, direccion);
 this.profesion = profesion;
 this.correo = correo;
 this.contrasenia = contrasenia;
}**public Docente(String cedula, String nombre, String apellido, int edad, String direccion) {**
 super(cedula, nombre, apellido, edad, direccion);
}**public Docente(String profesion, String cedula, String nombre, String apellido, int edad, String direccion) {**
 super(cedula, nombre, apellido, edad, direccion);
 this.profesion = profesion;
}**public Curso getCurso() {**
 return curso;
}**public void setCurso(Curso curso) {**
 this.curso = curso;
}**public String getProfesion() {**
 return profesion;
}**public void setProfesion(String profesion) {**
 this.profesion = profesion;
}**public String getCorreo() {**
 return correo;

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

```

}

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

public String getContrasenia() {
    return contrasenia;
}

public void setContrasenia(String contrasenia) {
    this.contrasenia = contrasenia;
}

@Override
public String toString() {
    return "Docente{" + "profesion=" + profesion + ", correo=" + correo + ", contrasenia=" + contrasenia +
}";
}

}

Paquete ec.edu.ups.controlador
Clase Abstracta Controlador
package ec.edu.ups.controlador;

import java.io.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

/**
 *
 * @author Anahi
 */
public abstract class Controlador<E> {
    private List<E> lista;

    private String ruta;

    public Controlador(String ruta) {
        lista = new ArrayList();
        this.ruta = ruta;
    }

    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("True");
        return lista.remove(objetoE);
    }
    System.out.println("False");
}
```

```
        return false;
    }

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

Clase ControladorPersona

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

import ec.edu.ups.modelo.Persona;

/**
 *
 * @author Anahi
 */
public class ControladorPersona extends Controlador<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;
}

}

```

ControladorAlumno


```

package ec.edu.ups.controlador;

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

/**
 *
 * @author Anahi
 */

```


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

```

public class ControladorAlumno extends Controlador<Alumno> {

    private ControladorPersona controlador;

    public ControladorAlumno(ControladorPersona controlador, String ruta) {
        super(ruta);
        this.controlador = controlador;
    }
    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;

    }

    @Override
    public boolean validar(Alumno objeto) {
        return controlador.validar(objeto);
    }

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

}

```

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 Anahi
 */
public class ControladorActividad extends Controlador<Actividad> {

```

```
private Pattern patron;
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) {
    this.patron =
Pattern.compile("<a\\shref=\\\"\\\"/store\\\"/apps\\\"/details\\\"?id=\"
+ \"(\\w+) (\\.(\\w+))+\\\"(\\s)?<div\\sclass=\\\"\\\"\"
+ \"\"
+ \"((\\w+)\\s(\\w+))\\\"\\\"stitle=\\\"\\\"(\\w+|([.,\\\"/#!$%^&\\\";:=
_\\s]))*\\\">\"");
}

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.getLista().iterator();
while (i.hasNext()) {
    actividad = (Actividad) i.next();
    lista.add(actividad);
}
return lista;
}
}
```

CotroladorCurso

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

```
import ec.edu.ups.modelo.Curso;
```

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

```
import java.util.Iterator;
```

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

```
/**
```

```
*
```

```
* @author Anahi
```

```
*/
```

```
public class ControladorCurso extends Controlador<Curso> {
```

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

```
    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;
    }
```

```
}
```

```
@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;
    }
}

}

ControladorRector
package ec.edu.ups.controlador;

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

/**
 *
 * @author Anahi
 */
public class ControladorRector extends Controlador<Rector> {

    private Rector rector;

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

    public Rector getRector() {
        return rector;
    }
}
```

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

@Override
public boolean validar(Rector objeto) {

    String correo = objeto.getCorreo();
    String pwd = objeto.getContrasenia();

    for (Rector rec : super.getLista()) {
        Rector r = (Rector) rec;
        if (r.getCorreo().equals(correo) && r.getContrasenia().equals(pwd)) {
            this.rector = r;
            return rector != null;
        }
    }
    return false;
}


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

public List<Rector> usuarios() {

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

public Rector buscarRector(Docente docente) {

    for (Rector rec : super.getLista()) {
        Rector r = (Rector) rec;
```

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

```

        if (docente.equals(r.getDocente())) {
            this.rector = r;
        }
    }
    return null;
}

public boolean iniciarSesion(String correo, String pass) {

    for (Rector rec : super.getLista()) {
        Rector r = (Rector) rec;
        if (r.getCorreo().equals(correo) && r.getContrasenia().equals(pass)) {
            this.rector = r;
            return true;
        }
    }
    return false;
}

}

ControladorDocente
package ec.edu.ups.controlador;

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

/**
 *
 * @author Anahi
 */
public class ControladorDocente extends Controlador<Docente> {


    private ControladorPersona controlador;

    public ControladorDocente(ControladorPersona controlador, String ruta) {
        super(ruta);
        this.controlador = controlador;
    }

    public List<Docente> docentes() {

        List<Docente> lista = new ArrayList();
        Docente docente;
        Iterator i = super.getLista().iterator();
        while (i.hasNext()) {

```

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

```

        docente = (Docente) i.next();
        lista.add(docente);

    }
    return lista;

}

@Override
public boolean validar(Docente objeto) {
    return controlador.validar(objeto);

}

@Override
public int generarId() {
    return 0;

}

}

```

2. Patrón de Diseño Utilizado

Iterator: El patrón de diseño iterator nos permite acceder secuencialmente a las listas por lo que hemos generado este patrón para las diferentes listas del código.

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:

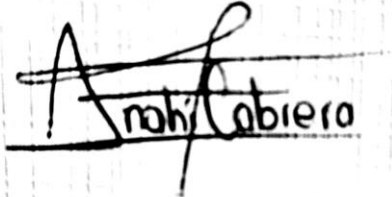
Aprenden a trabajar en grupo dentro de plazos de tiempo establecidos, manejando el lenguaje de programación de Java.

RECOMENDACIONES:

Realizar el trabajo dentro del tiempo establecido.

Nombre de estudiante: Cabrera Bermeo Edith Anahí

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



Firma de estudiante: _____