GUÍA DE LABORATORIO 3

Programación Orientada a ObjetosHERENCIA

NOMBRES: LAURA VALENTINA VARGAS SIERRA
IVONE GISELA LOPEZ CRUZ
LINK GIT: https://github.com/IvoneLopez2407/Lab-Herencia.git

Objetivo:

Aplicar los principales conceptos de herencia para crear aplicaciones evidenciando del potencial de los mecanismos ofrecidos por la orientación a objetos:

Preparación y creación del proyecto:

1. Considerando los conceptos vistos de herencia, cree un nuevo proyecto con las siguientes Clases, depúrelo y ejecútelo. Luego analice el ejemplo dado y responda el cuestionario adjunto.

//Implementación de la clase genérica Persona

```
package herencia;
import javax.swing.JOptionPane;
/**
* Clase Java para implementar una superclase padre
* que maneja los datos de un objeto de tipo persona
* de forma genérica
*/
public class Persona
{
/***********************************/
/***********************************/
//Atributos de la clase
protected String nombre;
protected String apellido;
protected int edad;
protected double peso;

    //Constructor de la clase
    public Persona() {
    // TODO Auto-generated constructor stub
    }
    //Constructor sobrecargado
    public Persona (String nombre, String apellido, int edad, double
    peso)
    {
    this.nombre = nombre;
    this.apellido = apellido;
    this.edad = edad;
    this.peso = peso;
    }

    //Métodos de Encapsulamiento
    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 double getPeso() {
return peso;
}
public void setPeso(double peso) {
this.peso = peso;
}

//Métodos genéricos para un objeto de tipo persona
//Método para mostrar los datos de identificación de la persona
public void imprimirDatosPersona ()
{
String datosPersona = "";
datosPersona = "Nombre " + nombre + "\n"
+ "Apellido: " + apellido + "\n" +
"Edad: " + edad + "\n" +
"Peso: " + peso;
JOptionPane.showMessageDialog(null,datosPersona);
definirEstadoPersona(edad);
}
//Método para definir el estado de la persona

    public void definirEstadoPersona (int edad)
{
String estado = "";
if (edad < 18)
{
estado = "Menor de edad";
}
else
{
estado = "Mayor de edad";
}
JOptionPane.showMessageDialog(null,"La persona" + nombre
+ " " + apellido + " es " + estado);
}
//Método para leer un entero (int)
public int leerDatoTipoEntero (String mensaje)
{
int valor = 0;
try
{
Valor=Integer.parseInt(JOptionPane.showInputDial
og (null,mensaje));
```

```
        return (valor);
        }
        catch (Exception errorIngreso)
        {
        errorIngreso.printStackTrace();
        return (0);
        }
        }
        //Método para leer un double
        public double leerDatoTipoReal (String mensaje)
        {
        double valor = 0.0;
        valor =
        Double.parseDouble
        (JOptionPane.showInputDialog(null,mensaje));
        return (valor);
        }
        //Método para leer un booleano
        public boolean leerDatoTipoBooleano (String mensaje)
        {
        boolean valor = false;
        valor =
        Boolean.parseBoolean
        (JOptionPane.showInputDialog(null,mensaje));
        return (valor);
        }
        public String leerDatoTipoCadena (String mensaje)
        {
        String valor = "";
        valor = JOptionPane.showInputDialog(null,mensaje);
        return (valor);
        }
}
```

//Implementación de la clase para definir un empleado

```
package herencia;
import javax.swing.JOptionPane;
/**
* Clase Java para implementar una subclase
* de tipo empleado que muestra el comportamiento
* propio de un objeto de este tipo
*/
public class Empleado extends Persona
{
//Atributos de la subclase
private double salario;
private String codigoLaboral;
private boolean esEmpleadoFijo;
        //Constructor de la clase
        public Empleado()
        {
        // TODO Auto-generated constructor stub
        }

        //Constructor de la subclase
        public Empleado (String nombre, String apellido, int edad, int peso,
        double salario, String codigoLaboral, boolean esEmpleadoFijo)
```

```
{

//Usamos el operador super para establecer el constructor
//de la superclase
super(nombre, apellido, edad, peso);

//Definimos los atributos propios de este objeto que hereda
this.salario = salario;
this.codigoLaboral = codigoLaboral;
this.esEmpleadoFijo = esEmpleadoFijo;
}

//Métodos de encapsulamiento
public double getSalario() {
        return salario;
}
public void setSalario(double salario) {
        this.salario = salario;
}
public String getCodigoLaboral() {
        return codigoLaboral;
}
public void setCodigoLaboral(String codigoLaboral) {
        this.codigoLaboral = codigoLaboral;
}
public boolean isEsEmpleadoFijo() {
        return esEmpleadoFijo;
}
public void setEsEmpleadoFijo(boolean esEmpleadoFijo) {
        this.esEmpleadoFijo = esEmpleadoFijo;
}
//Métodos propios del empleado

//Método para calcular el salario del empleado
public double calcularSalario (boolean esEmpleadoFijo, double valorHora, int numeroHorasTrabajadas)
{
double salario = 0.0;
double bono = 0.0;
if (esEmpleadoFijo == true)
{
bono = 5200.00;
}
else
{
bono = 1200.00;
}
salario = numeroHorasTrabajadas * valorHora + bono;
return (salario);
}
//Método para imprimir el sueldo del empleado
public void imprimirAsignacionSalario (String nombre, doublesueldo)
{
JOptionPane.showMessageDialog(null,"El empleado " + nombre +" recibe un salario de " + sueldo);
}
//Método para ingresar los datos del empleado
public Empleado ingresarDatosEmpleado ()
{
        Empleado nuevoEmpleado = new Empleado();
        String nombre = "";
        String apellido = "";
```

```
int edad = 0;
double peso = 0.0;
int numeroHoras = 0;
double valorHora = 0.0;
double salario = 0.0;
boolean esEmpleadoFijo = false;
String codigo = "";
codigo = leerDatoTipoCadena("Ingrese el código del empleado: ");
nombre = leerDatoTipoCadena("Ingrese el nombre del empleado: ");
apellido = leerDatoTipoCadena("Ingrese el apellido del empleado: ");
edad = leerDatoTipoEntero("Ingrese la edad del empleado: ");
peso = leerDatoTipoReal("Ingrese el peso del empleado: ");

//Obtenemos los datos que no hacen parte de la clase
numeroHoras = leerDatoTipoEntero("Ingrese el número de horas trabajadas por el empleado: ");
valorHora = leerDatoTipoReal("Ingrese el valor de la hora: ");
esEmpleadoFijo = leerDatoTipoBooleano("¿El empleado es fijo (true/false)?: ");

//Calculamos el salario
salario = calcularSalario(esEmpleadoFijo, valorHora,numeroHoras);
//Definimos los datos para el constructor
nuevoEmpleado.setNombre(nombre);
nuevoEmpleado.setApellido(apellido);
nuevoEmpleado.setCodigoLaboral(codigo);
nuevoEmpleado.setEdad(edad);
nuevoEmpleado.setPeso(peso);
nuevoEmpleado.setSalario(salario);
nuevoEmpleado.setEsEmpleadoFijo(esEmpleadoFijo);

//Retornamos el objeto
return (nuevoEmpleado);
}
```

//Método para imprimir el reporte de status del empleado

```
public void imprimirReporteEstadoEmpleado()
{
//Usamos nuevamente la herencia para llamar los métodos
//que imprimen los datos del empleado
imprimirDatosPersona();
//Ahora imprimimos los datos propios del estudiante
JOptionPane.showMessageDialog(null,"El código laboral del empleado es: " + codigoLaboral);
//Asignación del salario
imprimirAsignacionSalario(nombre,salario);
if (esEmpleadoFijo == true)
{
JOptionPane.showMessageDialog(null,"El empleado es un
trabajador fijo de la empresa");
}
else
{
JOptionPane.showMessageDialog(null,"El empleado es un
trabajador contratado de la empresa");
}
}
//Método sobreescrito para definir el estado del empleado
public void definirEstadoPersona (int edad)
{
String estado = "";
if (edad < 21)
{
```

```
estado = "Menor de edad legal";
}
else
{
estado = "Mayor de edad legal";
}
JOptionPane.showMessageDialog(null,"La persona " + nombre + "
" + apellido + " es " + estado);
}
}
```

//CLASE SISTEMA INFORMACION

```
package herencia;
import javax.swing.JOptionPane;
/**
* Clase Java para implementar un cliente
* que emplea la jerarquía de herencia de la persona,
* el empleado y el estudiante para mostrar los cálculos
* de cada uno de los respectivos procesos implementados
* por cada clase
*/
public class SistemaInformacion
{

	//Constructor de la clase
	public SistemaInformacion() {
	// TODO Auto-generated constructor stub
	}
	/**
	* @param args
	*/
	public static void main(String[] args)
	{

	// TODO Auto-generated method stub
	int opcion = 0;
	Persona sistemaInformacion = new Persona();
	try
	{
	//Usamos la herencia de la superclase para acceder a los
	//métodos de lectura de datos
	do

	opcion = sistemaInformacion.leerDatoTipoEntero("Ingrese la opción 1 para procesar los datos del
	empleado, 3 para salir: ");
	if (opcion == 1)
	{
	JOptionPane.showMessageDialog(null,"Procesando datos del empleado");
	Empleado manejadorEmpleado = new Empleado();
	//Leemos los datos
	manejadorEmpleado =manejadorEmpleado.ingresarDatosEmpleado();
	//Mostramos el reporte de datos
	manejadorEmpleado.imprimirReporteEstadoEmpleado();
	}
```

```
else
if (opcion == 3)
{
JOptionPane.showMessageDialog(null,"Saliendo del Sistema");
break;
}
else
{
JOptionPane.showMessageDialog(null,"Opción inválida");
}
} while (opcion <= 1 || opcion >= 3);//fin del do-while
}//Fin del try
catch (Exception errorMain)
{
JOptionPane.showMessageDialog(null,"Error en la digitación: ");
errorMain.printStackTrace();
}
}
}
```

1. Que métodos de la Superclase son utilizados en la subclase:

   Los métodos de la Superclase Persona que son utilizados en la subclase Empleado son:
   - El método constructor de la superclase: super (nombre, apellido, edad, peso);
   - Los métodos getNombre(), getApellido(), getEdad(), getPeso().
   - El método imprimirDatosPersona().
   - Los métodos leerDatoTipoEntero(), leerDatoTipoReal(), leerDatoTipoBooleano(), leerDatoTipoCadena().

2. Que función cumple el operador Super:

   El operador super tiene como función invocar desde la subclase Empleado al constructor de la superclase Persona, de esta manera se inicializa los atributos heredados de la superclase.

3. Observe los atributos de la Clase Padre, ¿cuál es su visibilidad? ¿De qué manera se hace uso de estos en la clase hija?

   La visibilidad de los atributos de la clase padre es protected. La clase hija hace uso de estos atributos por medio de los diferentes métodos públicos como setNombre, setApellido, setEdad, setPeso, que establece los valores de los atributos de la clase Padre, y se obtengan por medio de los métodos getNombre, getApellido, getEdad, getPeso.

4. Describa cómo funciona el método imprimirReporteEstadoEmpleado():

   El método funciona de tal manera que imprime el Estado del Empleado:
   - Llama al método imprimirDatosPersona(); que imprime los valores de los atributos de la superclase.
   - Llama al método imprimirAsignacionSalario(nombre,salario); el cual imprime el salario del Empleado.
   - Por medio de la estructura if, verifica si el empleado es fijo o no, y dependiendo de ello imprime un mensaje informando si este es un empleado fijo o contratado.

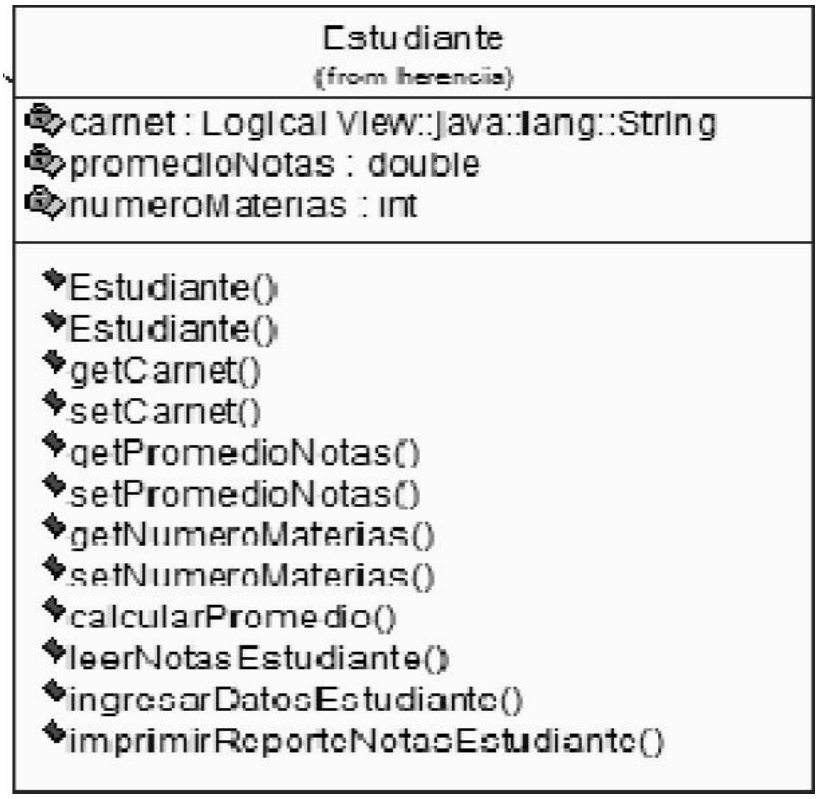
5. Que objetos se instancian en la Clase **SistemaInformacion.**

   Se instancia un objeto de tipo Empleado

6. Compare el método definirEstadoPersona (int edad) con el mismo método sobrescrito para definir el estado del empleado. ¿Qué diferencias tienen? ¿Por qué se hace?

La clase Persona tiene un método definirEstadoPersona() así como también lo tiene la subclase Empleado, este toma un parámetro de edad y retorna un estado basado en la edad. En el caso de la clase Persona, señala si la persona es menor o mayor de edad, mientras que en la subclase Empleado señala si es Menor de edad legal o mayor de edad legal. Sin embargo, el método definirEstadoPersona() en la subclase Empleado esta sobrescrito para incluir el estado del empleado.

7. Cree una nueva subclase llamada Estudiante según el siguiente diagrama de clases:



El sistema debe solicitar los datos del estudiante, el número de materias que el estudiante cursa, solicitar la nota de cada una de estas materias y calcular el promedio aritmético.

Agregue una opción de menú dentro de la clase SistemaInformacion que Lea los datos del Estudiante e imprima el reporte de notas (datos del estudiante (usando herencia), carnet, numero de materias y promedio.

Copie el código fuente de la nueva clase en este informe.

CODIGO FUENTE DE LA NUEVA CLASE ESTUDIANTE:

```
package com.mycompany.sistemainformacion;
import javax.swing.JOptionPane;

/**
 * @author Valentina Vargas e Ivone Lopez
 */
public class Estudiante extends Persona {

    //ATRIBUTOS DE LA SUBCLASE ESTUDIANTE
    private String carnet;
    private double promedioNotas;
    private int numeroMaterias;
```

```
    //CONSTRUCTOR DE LA CLASE
    public Estudiante() {
    }
    //CONSTRUCTOR DE LA SUBCLASE
    public Estudiante(String nombre, String apellido, int edad, double peso, String carnet, int numeroMaterias) {
        super(nombre, apellido, edad, peso);
        this.carnet = carnet;
        this.numeroMaterias = numeroMaterias;
        this.promedioNotas = 0.0;
    }

    //METODOS DE ENCAPSULAMIENTO
    public String getCarnet() {
        return carnet;
    }
    public void setCarnet(String carnet) {
        this.carnet = carnet;
    }
    public double getPromedioNotas() {
        return promedioNotas;
    }
    public void setPromedioNotas(double promedioNotas) {
        this.promedioNotas = promedioNotas;
    }
    public int getNumeroMaterias() {
        return numeroMaterias;
    }
    public void setNumeroMaterias(int numeroMaterias) {
        this.numeroMaterias = numeroMaterias;
    }

    //METODOS PROPIOS DEL ESTUDIANTE
    public void calcularPromedioNotas(double[] notas) {
        double suma = 0;
        for (double nota : notas) {
            suma = suma + nota;
        }
        this.promedioNotas = suma / notas.length;
    }

    public void imprimirReporteEstudiante() {
    //Usamos nuevamente la herencia para llamar los métodos que imprimen los datos del empleado
    imprimirDatosPersona();
    //Ahora imprimimos los datos propios del estudiante
    JOptionPane.showMessageDialog(null,"Carnet:  " +carnet);
    JOptionPane.showMessageDialog(null,"Numero de materias:  " + promedioNotas);
    JOptionPane.showMessageDialog(null,"Promedio:  " + promedioNotas);
    }
}
```