

Informe Técnico: Sistema de Gestión de Empleados

Luis Carlos Cortez Guzmán

Introducción

El presente informe detalla la creación de un sistema para gestionar diferentes tipos de empleados en una empresa, utilizando la programación orientada a objetos en Java. El sistema maneja empleados de tiempo completo, por horas y temporales, cada uno con características y beneficios específicos.

Estructura del Programa

El programa está compuesto por una clase padre `Empleado`` y tres clases hijas: `EmpleadoTiempoCompleto``, `EmpleadoPorHoras`` y `EmpleadoTemporal``. Cada clase hija extiende la funcionalidad de la clase padre añadiendo atributos y métodos específicos.

Clase Padre: Empleado

La clase `Empleado`` contiene los atributos y métodos comunes a todos los tipos de empleados.

Atributos:

- `nombre`` (String): Nombre del empleado.
- `id`` (int): Identificador único del empleado.
- `salario`` (double): Salario base del empleado.

Métodos:

- Constructor: Inicializa los atributos `nombre``, `id`` y `salario``.
- Getters y Setters: Métodos para acceder y modificar cada uno de los atributos.
- `calcularSalarioFinal()`: Método que devuelve el salario final (será sobrescrito en las subclases).
- `mostrarDetalles()`: Método que devuelve la información básica del empleado.

```

Empleados > J Empleado.java
1  package Empleados;
2
3  public class Empleado {
4      private String nombre;
5      private int id;
6      private double salario;
7
8      // Constructor
9  > public Empleado(String nombre, int id, double salario) { ...
14
15      // Getters y Setters
16  > public String getNombre() { ...
19
20  > public void setNombre(String nombre) { ...
23
24  > public int getId() { ...
27
28  > public void setId(int id) { ...
31
32  > public double getSalario() { ...
35
36  > public void setSalario(double salario) { ...
39
40      // Método para calcular el salario final (será sobrescrito en las subclases)
41  > public double calcularSalarioFinal() { ...
44
45      // Método para mostrar detalles del empleado
46  > public String mostrarDetalles() { ...
49  }

```

EmpleadoTiempoCompleto

Atributo adicional:

- `bonificacionAnual` (double): Bonificación anual del empleado.

Métodos:

- Constructor: Inicializa los atributos de la clase padre y el atributo `bonificacionAnual`.
- calcularSalarioFinal(): Incluye la bonificación anual en el cálculo del salario final.
- mostrarDetalles(): Incluye la bonificación anual en la información mostrada.

```

Empleados > J EmpleadoTiempoCompleto.java > ...
1  package Empleados;
2
3  public class EmpleadoTiempoCompleto extends Empleado{
4      private double bonificacionAnual;
5
6      // Constructor
7  > public EmpleadoTiempoCompleto(String nombre, int id, double salario, double bonificacionAnual) { ...
11
12      // Getter y Setter para bonificación anual
13  > public double getBonificacionAnual() { ...
16
17  > public void setBonificacionAnual(double bonificacionAnual) { ...
20
21      // Override de calcularSalarioFinal
22      @Override
23      public double calcularSalarioFinal() {
24          return getSalario() + bonificacionAnual;
25      }
26
27      // Override de mostrarDetalles
28      @Override
29      public String mostrarDetalles() {
30          return super.mostrarDetalles() + "\nBonificación Anual: " + bonificacionAnual;
31      }
32  }

```

EmpleadoPorHoras

Atributos adicionales:

- `horasTrabajadas` (int): Número de horas trabajadas.
- `tarifaPorHora` (double): Tarifa por hora trabajada.

Métodos:

- Constructor: Inicializa los atributos de la clase padre, `horasTrabajadas` y `tarifaPorHora`.
- calcularSalarioFinal(): Calcula el salario en base a las horas trabajadas y la tarifa por hora.
- mostrarDetalles(): Incluye las horas trabajadas y la tarifa por hora en la información mostrada.

```

Empleados > J EmpleadoPorHoras.java > ...
1  package Empleados;
2
3  public class EmpleadoPorHoras extends Empleado{
4      private int horasTrabajadas;
5      private double tarifaPorHora;
6
7      // Constructor
8  > public EmpleadoPorHoras(String nombre, int id, double salario, int horasTrabajadas, double tarifaPorHora) { ...
13
14      // Getters y Setters para horas trabajadas y tarifa por hora
15  > public int getHorasTrabajadas() { ...
18
19  > public void setHorasTrabajadas(int horasTrabajadas) { ...
22
23  > public double getTarifaPorHora() { ...
26
27  > public void setTarifaPorHora(double tarifaPorHora) { ...
30
31      // Override de calcularSalarioFinal
32      @Override
33  > public double calcularSalarioFinal() { ...
36
37      // Override de mostrarDetalles
38      @Override
39      public String mostrarDetalles() {
40          return super.mostrarDetalles() + "\nHoras Trabajadas: " + horasTrabajadas + "\nTarifa por Hora: " + tarifaPorHora;
41      }
42  }

```

EmpleadoTemporal

Atributos adicionales:

- `fechaInicio` (String): Fecha de inicio del contrato.
- `fechaFin` (String): Fecha de fin del contrato.

Métodos:

- Constructor: Inicializa los atributos de la clase padre, `fechaInicio` y `fechaFin`.
- calcularSalarioFinal(): Considera un bono al finalizar el contrato.
- mostrarDetalles(): Incluye la fecha de inicio y fin en la información mostrada.

```

Empleados > J EmpleadoTemporal.java > ...
1  package Empleados;
2
3  public class EmpleadoTemporal extends Empleado{
4      private String fechaInicio;
5      private String fechaFin;
6
7      // Constructor
8  > public EmpleadoTemporal(String nombre, int id, double salario, String fechaInicio, String fechaFin) {...
13
14      // Getters y Setters para fecha de inicio y fin
15  > public String getFechaInicio() {...
18
19  > public void setFechaInicio(String fechaInicio) {...
22
23  > public String getFechaFin() {...
26
27  > public void setFechaFin(String fechaFin) {...
30
31      // Override de calcularSalarioFinal (considerando un bonus al finalizar el contrato)
32      @Override
33  > public double calcularSalarioFinal() {...
37
38      // Override de mostrarDetalles
39      @Override
40  > public String mostrarDetalles() {...
43  }

```

Clase Principal Main

La clase `Main` se utiliza para crear instancias de cada tipo de empleado y mostrar sus detalles y salario final.

```

J Main.java > ...
1  import Empleados.*;
2  public class Main {
3      public static void main(String[] args) {
4          EmpleadoTiempoCompleto empTiempoCompleto = new EmpleadoTiempoCompleto(nombre:"Luis", id:1, salario:2500, bonificacionAnual:5000);
5          EmpleadoPorHoras empPorHoras = new EmpleadoPorHoras(nombre:"Carlos", id:2, salario:0, horasTrabajadas:100, tarifaPor_20);
6          EmpleadoTemporal empTemporal = new EmpleadoTemporal(nombre:"Cortez", id:3, salario:2000, fechaInicio:"2024-01-01", fechaFin:"2024-01-01");
7
8          System.out.println(empTiempoCompleto.mostrarDetalles());
9          System.out.println("Salario Final: " + empTiempoCompleto.calcularSalarioFinal());
10         System.out.println();
11
12         System.out.println(empPorHoras.mostrarDetalles());
13         System.out.println("Salario Final: " + empPorHoras.calcularSalarioFinal());
14         System.out.println();
15
16         System.out.println(empTemporal.mostrarDetalles());
17         System.out.println("Salario Final: " + empTemporal.calcularSalarioFinal());
18     }
19 }

```

Explicación de la Ejecución del Programa

1. Instanciación de Objetos:

- Se crean instancias de ``EmpleadoTiempoCompleto``, ``EmpleadoPorHoras`` y ``EmpleadoTemporal`` utilizando los constructores definidos en cada clase.

- Ejemplo: ``EmpleadoTiempoCompleto empTiempoCompleto = new EmpleadoTiempoCompleto("Carlos López", 1, 50000, 5000);``

2. Llamada a Métodos:

- Se llama al método ``mostrarDetalles()`` para cada instancia creada.

- Ejemplo: ``System.out.println(empTiempoCompleto.mostrarDetalles());``

3. Salida del Programa:

- El método ``mostrarDetalles()`` de cada clase hija sobrescribe el método de la clase padre ``Empleado`` para incluir información específica (bonificaciones, horas trabajadas, tarifas, fechas).

- Se imprime la información detallada de cada empleado, mostrando el nombre, ID, salario, y los atributos adicionales según el tipo de empleado.

- También se llama al método ``calcularSalarioFinal()`` para calcular y mostrar el salario final considerando las bonificaciones y horas trabajadas.

Ejemplo de Salida

```
Nombre: Luis
ID: 1
Salario: 2500.0
Bonificación Anual: 5000.0
Salario Final: 7500.0

Nombre: Carlos
ID: 2
Salario: 0.0
Horas Trabajadas: 100
Tarifa por Hora: 20.0
Salario Final: 2000.0

Nombre: Cortez
ID: 3
Salario: 2000.0
Fecha de Inicio: 2024-01-01
Fecha de Fin: 2024-06-30
Salario Final: 2200.0
```

Conclusión

Este sistema de gestión de empleados permite manejar eficientemente la información de diferentes tipos de empleados, utilizando la herencia y el polimorfismo en Java. La estructura del programa es extensible, lo que facilita la adición de nuevos tipos de empleados en el futuro si es necesario.