Tecnicatura Universitaria en Programación a Distancia

PROGRAMACIÓN II

Trabajo Práctico 4: Programación Orientada a Objetos

Alumno: Durussel Federico
Comisión 13

Caso práctico:

Sistema de Gestión de Empleados

Modelar una clase Empleado que represente a un trabajador en una empresa.
Esta clase debe incluir constructores sobrecargados, métodos sobrecargados y el uso de atributos aplicando encapsulamiento y métodos estáticos para llevar control de los objetos creados.

Clase Main:

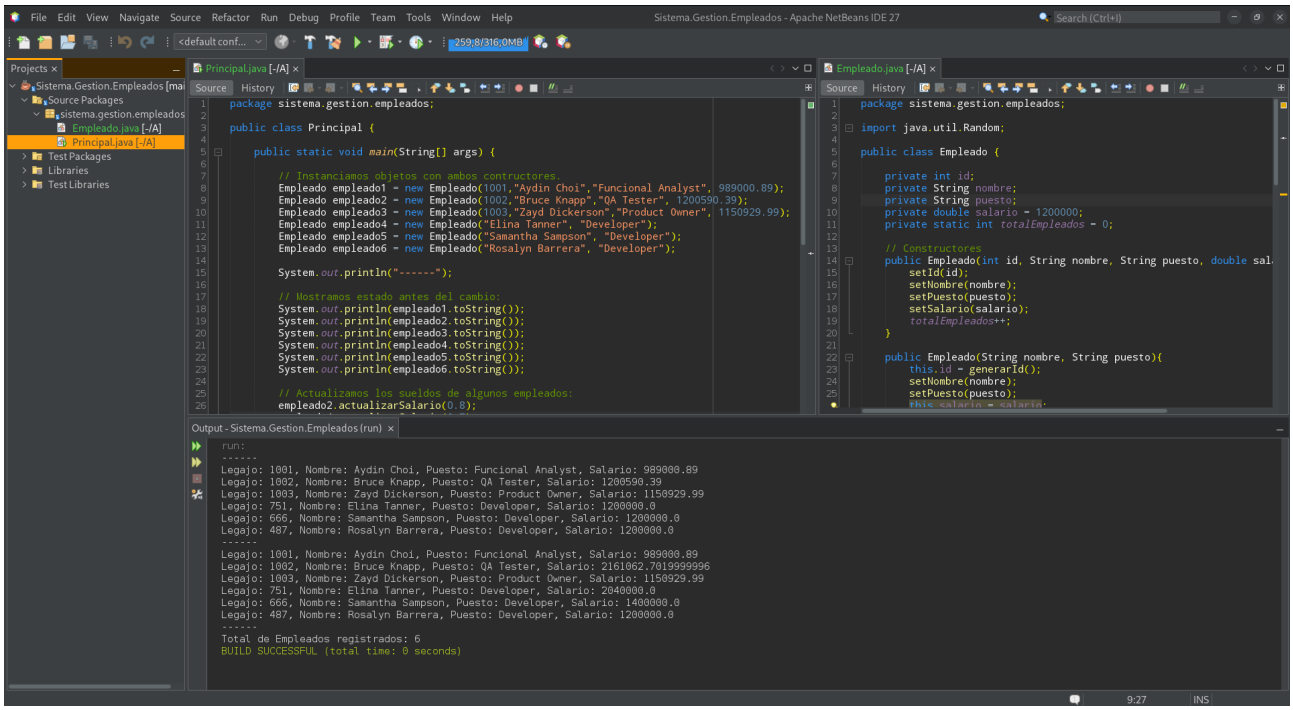
```
package sistema.gestion.empleados;

public class Principal {

    public static void main(String[] args) {

        // Instanciamos objetos con ambos contructores.
        Empleado empleado1 = new Empleado(1001,"Aydin Choi","Funcional Analyst", 989000.89);
        Empleado empleado2 = new Empleado(1002,"Bruce Knapp","QA Tester", 1200590.39);
        Empleado empleado3 = new Empleado(1003,"Zayd Dickerson","Product Owner", 1150929.99);
        Empleado empleado4 = new Empleado("Elina Tanner", "Developer");
        Empleado empleado5 = new Empleado("Samantha Sampson", "Developer");
        Empleado empleado6 = new Empleado("Rosalyn Barrera", "Developer");

        System.out.println("------");

        // Mostramos estado antes del cambio:
        System.out.println(empleado1.toString());
        System.out.println(empleado2.toString());
        System.out.println(empleado3.toString());
        System.out.println(empleado4.toString());
        System.out.println(empleado5.toString());
        System.out.println(empleado6.toString());

        // Actualizamos los sueldos de algunos empleados:
        empleado2.actualizarSalario(0.8);
        empleado4.actualizarSalario(0.7);
        empleado5.actualizarSalario(200000);

        System.out.println("------");

        // Volvemos a mostrar estado:
        System.out.println(empleado1.toString());
        System.out.println(empleado2.toString());
        System.out.println(empleado3.toString());
        System.out.println(empleado4.toString());
        System.out.println(empleado5.toString());
        System.out.println(empleado6.toString());

        System.out.println("------");

        // Mostramos el total de empleados:
        Empleado.mostrarTotalEmpleados();

    }
}
```

Clase Empleado

(continua en la próxima página)

```java
package sistema.gestion.empleados;

import java.util.Random;

public class Empleado {

    private int id;
    private String nombre;
    private String puesto;
    private double salario = 1200000;
    private static int totalEmpleados = 0;

    // Constructores
    public Empleado(int id, String nombre, String puesto, double salario) {
        setId(id);
        setNombre(nombre);
        setPuesto(puesto);
        setSalario(salario);
        totalEmpleados++;
    }

    public Empleado(String nombre, String puesto){
        this.id = generarId();
        setNombre(nombre);
        setPuesto(puesto);
        this.salario = salario;
        totalEmpleados++;
    }
    // Setters
    public void setId(int id){
        if (id > 0) {
            this.id = id;
        }
    }

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

    public void setPuesto(String puesto) {
        if (nombre != null) {
            this.puesto = puesto;
        }
    }

    public void setSalario(double salario) {
        if (salario > 0) {
            this.salario = salario;
        }
    }

    // Métodos
    private int generarId() {
        Random random = new Random();
        return random.nextInt(1000);
    }

    public double actualizarSalario(int aumento) {
        return this.salario = salario + aumento;
    }

    public double actualizarSalario(double aumento) {
        return this.salario = salario + (salario * aumento);
    }

    public static void mostrarTotalEmpleados(){
        System.out.println("Total de Empleados registrados: " + Empleado.totalEmpleados);
    }

    @Override
    public String toString(){
        return "Legajo: " + id + ", Nombre: " + nombre +
               ", Puesto: " + puesto + ", Salario: " + salario;
    }
}
```

Ejecución:




```
run:
------
Legajo: 1001, Nombre: Aydin Choi, Puesto: Funcional Analyst, Salario: 989000.89
Legajo: 1002, Nombre: Bruce Knapp, Puesto: QA Tester, Salario: 1200590.39
Legajo: 1003, Nombre: Zayd Dickerson, Puesto: Product Owner, Salario: 1150929.99
Legajo: 751, Nombre: Elina Tanner, Puesto: Developer, Salario: 1200000.0
Legajo: 666, Nombre: Samantha Sampson, Puesto: Developer, Salario: 1200000.0
Legajo: 487, Nombre: Rosalyn Barrera, Puesto: Developer, Salario: 1200000.0
------
Legajo: 1001, Nombre: Aydin Choi, Puesto: Funcional Analyst, Salario: 989000.89
Legajo: 1002, Nombre: Bruce Knapp, Puesto: QA Tester, Salario: 2161062.7019999996
Legajo: 1003, Nombre: Zayd Dickerson, Puesto: Product Owner, Salario: 1150929.99
Legajo: 751, Nombre: Elina Tanner, Puesto: Developer, Salario: 2040000.0
Legajo: 666, Nombre: Samantha Sampson, Puesto: Developer, Salario: 1400000.0
Legajo: 487, Nombre: Rosalyn Barrera, Puesto: Developer, Salario: 1200000.0
------
Total de Empleados registrados: 6
BUILD SUCCESSFUL (total time: 0 seconds)
```