Sistema de Gestión de Empleados

```
import java.util.*;

// Interfaz para evaluación de desempeño
interface Evaluable {
    void evaluarDesempeno(int puntuacion);
}

// Clase base Empleado
class Empleado implements Evaluable {
    protected String nombre;
    protected double salario;
    protected String puesto;

    public Empleado(String nombre, double salario, String puesto) {
        this.nombre = nombre;
        this.salario = salario;
        this.puesto = puesto;
    }

    public double calcularBono() {
        return salario * 0.05;
    }

    public void aumentarSalario(double porcentaje) {
        salario += salario * (porcentaje / 100);
        System.out.println(nombre + " ha recibido un aumento. Nuevo
salario: " + salario);
    }

    @Override
    public void evaluarDesempeno(int puntuacion) {
        System.out.println(nombre + " recibió una evaluación de " +
puntuacion + "/10");
        if (puntuacion >= 8) {
            aumentarSalario(10);
        }
    }
}

// Subclase Gerente
class Gerente extends Empleado {
    private String departamento;
    private List<Empleado> equipo;

    public Gerente(String nombre, double salario, String departamento) {
        super(nombre, salario, "Gerente");
        this.departamento = departamento;
        this.equipo = new ArrayList<>();
```

```
    }

    public void agregarEmpleado(Empleado empleado) {
        equipo.add(empleado);
        System.out.println(empleado.nombre + " ahora forma parte del
equipo de " + departamento);
    }

    @Override
    public double calcularBono() {
        return salario * 0.10;
    }
}

// Subclase Director
class Director extends Gerente {
    public Director(String nombre, double salario, String departamento) {
        super(nombre, salario, departamento);
    }

    @Override
    public double calcularBono() {
        return salario * 0.15;
    }

    public void aprobarPresupuesto() {
        System.out.println(nombre + " ha aprobado el presupuesto del
departamento.");
    }
}

// Clase Empresa
class Empresa {
    private List<Empleado> empleados;

    public Empresa() {
        empleados = new ArrayList<>();
    }

    public void contratarEmpleado(Empleado empleado) {
        empleados.add(empleado);
        System.out.println("Nuevo empleado contratado: " + empleado.nombre
+ " como " + empleado.puesto);
    }

    public void mostrarEmpleados() {
        System.out.println("\nLista de empleados:");
        for (Empleado e : empleados) {
            System.out.println(e.nombre + " - Puesto: " + e.puesto + " -
Salario: " + e.salario);
        }
    }
}
```

```
// Clase de prueba
public class TestGestionEmpleados {
    public static void main(String[] args) {
        Empresa empresa = new Empresa();

        Empleado emp1 = new Empleado("Juan", 3000, "Asistente");
        Gerente ger1 = new Gerente("Ana", 5000, "Ventas");
        Director dir1 = new Director("Carlos", 8000, "Finanzas");

        empresa.contratarEmpleado(emp1);
        empresa.contratarEmpleado(ger1);
        empresa.contratarEmpleado(dir1);

        ger1.agregarEmpleado(emp1);

        empresa.mostrarEmpleados();

        emp1.evaluarDesempeno(9);
        ger1.evaluarDesempeno(7);
        dir1.aprobarPresupuesto();
    }
}
```

Solución Ejercicio 1: Calculadora con sobrecarga

```
class Calculadora {
    int sumar(int a, int b) {
        return a + b;
    }

    double sumar(double a, double b) {
        return a + b;
    }

    int sumar(int a, int b, int c) {
        return a + b + c;
    }
}

public class CalculadoraTest {
    public static void main(String[] args) {
        Calculadora calc = new Calculadora();
        System.out.println("Suma de enteros: " + calc.sumar(5, 10));
        System.out.println("Suma de decimales: " + calc.sumar(5.5, 2.3));
        System.out.println("Suma de tres números: " + calc.sumar(1, 2,
3));
    }
}
```

Solución Ejercicio 2: Formateador de Texto

```
class FormateadorTexto {
    String formatear(String texto) {
        return texto.toUpperCase();
    }

    String formatear(String texto, boolean minusculas) {
        return minusculas ? texto.toLowerCase() : texto;
    }

    String formatear(String texto, int longitud) {
        return texto.length() > longitud ? texto.substring(0, longitud) :
texto;
    }
}

public class FormateadorTextoTest {
    public static void main(String[] args) {
        FormateadorTexto ft = new FormateadorTexto();
        System.out.println(ft.formatear("Hola Mundo"));
        System.out.println(ft.formatear("Hola Mundo", true));
        System.out.println(ft.formatear("Hola Mundo", 4));
    }
}
```

Solución Ejercicio 3: Vehículos y su velocidad

```
class Vehiculo {
    void mover(int velocidad) {
        System.out.println("El vehículo se mueve a " + velocidad + "
km/h");
    }

    void mover(double velocidad, String unidad) {
        System.out.println("El vehículo se mueve a " + velocidad + " " +
unidad);
    }
}

class Coche extends Vehiculo {
    @Override
    void mover(int velocidad) {
        System.out.println("El coche se mueve a " + velocidad + " km/h");
    }
}

public class VehiculoTest {
    public static void main(String[] args) {
```

```
        Vehiculo v = new Vehiculo();
        Coche c = new Coche();

        v.mover(60);
        v.mover(35.5, "millas por hora");
        c.mover(80);
    }
}
```

Ejercicio 4: Gestión de empleados

```
class Empleado {
    double calcularSalario() {
        return 2000.0;
    }
}

class Gerente extends Empleado {
    @Override
    double calcularSalario() {
        return super.calcularSalario() + 1000;
    }
}

class Programador extends Empleado {
    @Override
    double calcularSalario() {
        return super.calcularSalario() + 500;
    }
}

public class Main {
    public static void main(String[] args) {
        Empleado e = new Empleado();
        Gerente g = new Gerente();
        Programador p = new Programador();

        System.out.println("Salario Empleado: " + e.calcularSalario());
        System.out.println("Salario Gerente: " + g.calcularSalario());
        System.out.println("Salario Programador: " + p.calcularSalario());
    }
}
```

Ejercicio 5: Área de Figuras

```
class Figura {
    public double calcularArea(int lado) {
        return lado * lado;
```

```
    }

    public double calcularArea(int base, int altura) {
        return base * altura;
    }

    public double calcularArea(double radio) {
        return Math.PI * radio * radio;
    }
}

public class TestFigura {
    public static void main(String[] args) {
        Figura figura = new Figura();
        System.out.println("Área del cuadrado: " +
figura.calcularArea(4));
        System.out.println("Área del rectángulo: " +
figura.calcularArea(4, 5));
        System.out.println("Área del círculo: " +
figura.calcularArea(3.0));
    }
}
```

Ejercicio 6: Banco y tipos de cuentas

```
class CuentaBancaria {
    public double calcularInteres() {
        return 0;
    }
}

class CuentaAhorro extends CuentaBancaria {
    @Override
    public double calcularInteres() {
        return 3.0;
    }
}

class CuentaCorriente extends CuentaBancaria {
    @Override
    public double calcularInteres() {
        return 1.0;
    }
}

public class TestBanco {
    public static void main(String[] args) {
        CuentaBancaria ahorro = new CuentaAhorro();
        CuentaBancaria corriente = new CuentaCorriente();
        System.out.println("Interés en cuenta de ahorro: " +
ahorro.calcularInteres() + "%");
```

```
        System.out.println("Interés en cuenta corriente: " +
corriente.calcularInteres() + "%");
    }
}
```

Ejercicio 7: Animales y sonidos

```
class Animal {
    public void hacerSonido() {
        System.out.println("Sonido genérico");
    }
}

class Perro extends Animal {
    @Override
    public void hacerSonido() {
        System.out.println("Ladrido");
    }
}

class Gato extends Animal {
    @Override
    public void hacerSonido() {
        System.out.println("Maullido");
    }
}

public class TestAnimales {
    public static void main(String[] args) {
        Animal perro = new Perro();
        Animal gato = new Gato();
        perro.hacerSonido();
        gato.hacerSonido();
    }
}
```

Ejercicio 8: Envío de Mensajes

```
class Mensajero {
    public void enviarMensaje(String mensaje) {
        System.out.println("Mensaje: " + mensaje);
    }

    public void enviarMensaje(String mensaje, String destinatario) {
        System.out.println("Mensaje para " + destinatario + ": " +
mensaje);
    }

    public void enviarMensaje(String mensaje, String destinatario, boolean
```

```
urgente) {
        System.out.println("Mensaje para " + destinatario + " (Urgente: "
+ urgente + "): " + mensaje);
    }
}

public class TestMensajero {
    public static void main(String[] args) {
        Mensajero mensajero = new Mensajero();
        mensajero.enviarMensaje("Hola");
        mensajero.enviarMensaje("Hola", "Juan");
        mensajero.enviarMensaje("Hola", "Ana", true);
    }
}
```

Ejercicio 9: Cálculo de Descuentos

```
class Descuento {
    public double calcular(double precio, double porcentaje) {
        return precio - (precio * porcentaje / 100);
    }

    public double calcular(double precio, int cantidad, double porcentaje)
{
        double total = precio * cantidad;
        return total - (total * porcentaje / 100);
    }

    public double calcular(double precio, double porcentaje, boolean
clienteFrecuente) {
        double descuento = clienteFrecuente ? porcentaje + 5 : porcentaje;
        return precio - (precio * descuento / 100);
    }
}

public class TestDescuento {
    public static void main(String[] args) {
        Descuento descuento = new Descuento();
        System.out.println("Precio con descuento: " +
descuento.calcular(100, 10));
        System.out.println("Precio con descuento por cantidad: " +
descuento.calcular(100, 5, 10));
        System.out.println("Precio con descuento cliente frecuente: " +
descuento.calcular(100, 10, true));
    }
}
```

Ejercicio 10: Gestión de Pedidos

```
class Pedido {
    public void procesar() {
        System.out.println("Procesando pedido estándar");
    }
}

class PedidoExpress extends Pedido {
    @Override
    public void procesar() {
        System.out.println("Procesando pedido express - Entrega en 24
horas");
    }
}

class PedidoInternacional extends Pedido {
    @Override
    public void procesar() {
        System.out.println("Procesando pedido internacional - Tiempo de
entrega variable");
    }
}

public class TestPedidos {
    public static void main(String[] args) {
        Pedido pedido1 = new Pedido();
        Pedido pedido2 = new PedidoExpress();
        Pedido pedido3 = new PedidoInternacional();

        pedido1.procesar();
        pedido2.procesar();
        pedido3.procesar();
    }
}
```