

PRACTICA PROGRAMACION

1.CODIGO EN JAVA:

CLASE PERSONA:

```
1 package practica.de.programacion.ll;
2 public class Persona {
3     String nombre;
4     int edad;
5     String ciudad;
6
7     public Persona(String nombre, int edad, String ciudad) {
8         this.nombre = nombre;
9         this.edad = edad;
10        this.ciudad = ciudad;
11    }
12    //a)Agregar un metodo para mostrar el saludo:"Hola,soy{nombre}de{ciudad}"
13    //b)Crear tres personas y muestra su saludo
14    public void saludar() {
15        System.out.println("Hola, soy " + this.nombre + " de " + this.ciudad);
16    }
17    //c)Agregar un metodo para verificar si es mayor de edad
18    public boolean esMayorDeEdad() {
19        return this.edad >= 18;
20    }
21 }
```

PRINCIPAL:

```
1 package practica.de.programacion.ll;
2
3 public class PracticaDeProgramacionLl {
4
5     public static void main(String[] args) {
6         // a)Agregar un metodo para mostrar el saludo:"Hola,soy{nombre}de{ciudad}"
7         //b)Crear tres personas y muestra su saludo
8         Persona persona = new Persona("joel", 19, "LaPaz");
9         persona.saludar();
10
11        Persona persona2 = new Persona("Bray", 20, "Oruro");
12        persona2.saludar();
13
14        Persona persona3 = new Persona("hans", 15, "Tarija");
15        persona3.saludar();
16        //c)Agregar un metodo para verificar si es mayor de edad
17        System.out.println("--- Verificación de Mayor de Edad ---");
18        System.out.println(persona.nombre + " es mayor de edad: " + persona.esMayorDeEdad());
19        System.out.println(persona2.nombre + " es mayor de edad: " + persona2.esMayorDeEdad());
20        System.out.println(persona3.nombre + " es mayor de edad: " + persona3.esMayorDeEdad());
21    }
22 }
23 }
```

CORRIDA:

```
run:
Hola, soy joel de LaPaz
Hola, soy Bray de Oruro
Hola, soy hans de Tarija
--- Verificación de Mayor de Edad ---
joel es mayor de edad: true
Bray es mayor de edad: true
hans es mayor de edad: false
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

CLASE PERSONA:

```
ejercicio1.py > ...
1  class Persona:
2      def __init__(self, nombre, edad, ciudad):
3          self.nombre = nombre
4          self.edad = edad
5          self.ciudad = ciudad
6
7      # a) Agrega un método para mostrar el saludo: "Hola, soy {nombre} de {ciudad}"
8      def saludar(self):
9          print(f"Hola, soy {self.nombre} de {self.ciudad}")
10
11     # c) Agrega un método para verificar si es mayor de edad
12     def es_mayor_de_edad(self):
13         return self.edad >= 18
14
15     # b) Crea tres personas y muestra su saludo
16     persona1 = Persona("Joel", 18, "Cochabamba")
17     persona2 = Persona("Bray", 20, "Oruro")
18     persona3 = Persona("Hans", 15, "Santa Cruz")
19
20     print("--- Saludos de las personas ---")
21     persona1.saludar()
22     persona2.saludar()
23     persona3.saludar()
24
25     print("--- Verificación de Mayor de Edad ---")
26     print(f"{persona1.nombre} es mayor de edad: {persona1.es_mayor_de_edad()}")
27     print(f"{persona2.nombre} es mayor de edad: {persona2.es_mayor_de_edad()}")
28     print(f"{persona3.nombre} es mayor de edad: {persona3.es_mayor_de_edad()}")
29
```

CORRIDA:

```
esktop/Programacion 11/ejercicio1.py"
--- Saludos de las personas ---
Hola, soy Joel de Cochabamba
Hola, soy Bray de Oruro
Hola, soy Hans de Santa Cruz
--- Verificación de Mayor de Edad ---
Joel es mayor de edad: True
Bray es mayor de edad: True
Hans es mayor de edad: False
PS C:\Users\LENOVO\Desktop\Programacion 11>
```

3. CODIGO EN JAVA:

CLASE COCHE:

PRINCIPAL:

```
1 package practica.de.programacion.ll;
2
3 public class PracticaDeProgramacionLl {
4
5     public static void main(String[] args) {
6         // c) Crea dos coches
7         Coche coche1 = new Coche("Toyota", "Corolla");
8         Coche coche2 = new Coche("Nissan", "Sentra");
9
10        System.out.println("--- Estado inicial de los coches ---");
11        System.out.println(coche1.marca + " " + coche1.modelo + " - Velocidad: " + coche1.velocidad + " km/h");
12        System.out.println(coche2.marca + " " + coche2.modelo + " - Velocidad: " + coche2.velocidad + " km/h");
13
14        System.out.println("--- Acelerando los coches ---");
15        coche1.acelerar();
16        coche2.acelerar();
17        coche2.acelerar();
18
19        System.out.println("--- Frenando los coches ---");
20        coche1.frenar();
21        coche2.frenar();
22
23        System.out.println("--- Estado final de los coches ---");
24        System.out.println(coche1.marca + " " + coche1.modelo + " - Velocidad final: " + coche1.velocidad + " km/h");
25        System.out.println(coche2.marca + " " + coche2.modelo + " - Velocidad final: " + coche2.velocidad + " km/h");
26    }
27}
```

CORRIDA:

```
Página de Inicio × Salida - práctica de programación II (run) × PracticaDeProgramacionII.java × Cache  
run:  
--- Estado inicial de los coches ---  
Toyota Corolla - Velocidad: 20 km/h  
Nissan Sentra - Velocidad: 20 km/h  
--- Acelerando los coches ---  
Toyota Corolla acelera. Velocidad actual: 30 km/h  
Nissan Sentra acelera. Velocidad actual: 30 km/h  
Nissan Sentra acelera. Velocidad actual: 40 km/h  
--- Frenando los coches ---  
Toyota Corolla frena. Velocidad actual: 25 km/h  
Nissan Sentra frena. Velocidad actual: 35 km/h  
--- Estado final de los coches ---  
Toyota Corolla - Velocidad final: 25 km/h  
Nissan Sentra - Velocidad final: 35 km/h  
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

CLASE COCHE:

```
ejercicio2.py > ...
1  class Coche:
2      def __init__(self, marca, modelo):
3          self.marca = marca
4          self.modelo = modelo
5          self.velocidad = 10
6
7      # a) Agrega un método acelerar () que aumente la velocidad en 10
8      def acelerar(self):
9          self.velocidad += 10
10         print(f"{self.marca} {self.modelo} acelera. Velocidad actual: {self.velocidad} km/h")
11
12     # b) Agrega un método frenar () que disminuya la velocidad en 5
13     def frenar(self):
14         if self.velocidad >= 5:
15             self.velocidad -= 5
16             print(f"{self.marca} {self.modelo} frena. Velocidad actual: {self.velocidad} km/h")
17         else:
18             self.velocidad = 0
19             print(f"{self.marca} {self.modelo} frena hasta detenerse. Velocidad actual: {self.velocidad} km/h")
20
21     # c) Crea dos coches, aceléralos, frénalos y muestra sus velocidades
22     coche1 = Coche("Toyota", "Corolla")
23     coche2 = Coche("Nissan", "Sentra")
24
25     print("--- Estado inicial de los coches ---")
26     print(f"{coche1.marca} {coche1.modelo} - Velocidad: {coche1.velocidad} km/h")
27     print(f"{coche2.marca} {coche2.modelo} - Velocidad: {coche2.velocidad} km/h")
28
29     print("--- Acelerando los coches ---")
30     coche1.acelerar()
31     coche1.acelerar()
32     coche2.acelerar()
33
34     print("--- Frenando los coches ---")
35     coche1.frenar()
36     coche2.frenar()
37
38     print("--- Estado final de los coches ---")
39     print(f"{coche1.marca} {coche1.modelo} - Velocidad final: {coche1.velocidad} km/h")
40     print(f"{coche2.marca} {coche2.modelo} - Velocidad final: {coche2.velocidad} km/h")
```

CORRIDA:

```
PS C:\Users\LENOVO\Desktop\Programacion 11> & C:/User
PS C:\Users\LENOVO\Desktop\Programacion 11> & C:/User
--- Estado inicial de los coches ---
Toyota Corolla - Velocidad: 10 km/h
Nissan Sentra - Velocidad: 10 km/h
--- Acelerando los coches ---
Toyota Corolla acelera. Velocidad actual: 20 km/h
Toyota Corolla acelera. Velocidad actual: 30 km/h
Nissan Sentra acelera. Velocidad actual: 20 km/h
--- Frenando los coches ---
Toyota Corolla frena. Velocidad actual: 25 km/h
Nissan Sentra frena. Velocidad actual: 15 km/h
--- Estado final de los coches ---
Toyota Corolla - Velocidad final: 25 km/h
Nissan Sentra - Velocidad final: 15 km/h
PS C:\Users\LENOVO\Desktop\Programacion 11> █
```

5 CODIGO EN JAVA:

CLASE ESTUDIANTE:

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** Página de Inicio > Salida - práctica de programación II (run) > PracticaDeProgramacionII.java > Estudiante
- Toolbar:** Includes icons for Source, History, and various file operations.
- Code Editor:** Displays the following Java code:

```
1 package practica.de.programacion.ii;
2 public class Estudiante {
3     String nombre;
4     double nota_1;
5     double nota_2;
6
7     public Estudiante(String nombre, double nota_1, double nota_2) {
8         this.nombre = nombre;
9         this.nota_1 = nota_1;
10        this.nota_2 = nota_2;
11    }
12
13    // a) Agrega un método para calcular el promedio
14    public double calcularPromedio() {
15        return (this.nota_1 + this.nota_2) / 2.0;
16    }
17
18    // b) Agrega un método para verificar si aprobó (promedio >= 6)
19    public boolean aprobado() {
20        return calcularPromedio() >= 6.0;
21    }
22 }
```

PRINCIPAL:

CORRIDA ·

```
Página de Inicio  X  Salida - practica de programacion II (run)  X  PracticaDeProgramacionII.java

run:
--- Resultados de los estudiantes ---
--- Estudiante 1: joel ---
Promedio: 7.75
¿Aprobó?: Sí
--- Estudiante 2: Carlos ---
Promedio: 5.75
¿Aprobó?: No
--- Estudiante 3: Bay ---
Promedio: 9.5
¿Aprobó?: Sí
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

CLASE ESTUDIANTE:

```
ejercicio3.py > ...
1  class Estudiante:
2      def __init__(self, nombre, nota_1, nota_2):
3          self.nombre = nombre
4          self.nota_1 = nota_1
5          self.nota_2 = nota_2
6
7      # a) Agrega un método para calcular el promedio
8      def calcular_promedio(self):
9          return (self.nota_1 + self.nota_2) / 2
10
11     # b) Agrega un método para verificar si aprobó (promedio >= 6)
12     def aprobado(self):
13         return self.calcular_promedio() >= 6
14
15     # c) Crea tres estudiantes, muestra sus promedios y si aprobaron
16     estudiante1 = Estudiante("Joel", 7.5, 8.0)
17     estudiante2 = Estudiante("Carlos", 5.0, 6.5)
18     estudiante3 = Estudiante("Bray", 9.0, 9.5)
19
20     print("--- Resultados de los estudiantes ---")
21
22     # Muestra el promedio y si aprobaron para cada estudiante
23     print(f"--- Estudiante 1: {estudiante1.nombre} ---")
24     promedio1 = estudiante1.calcular_promedio()
25     print(f"Promedio: {promedio1}")
26     print(f"¿Aprobó?: {'Sí' if estudiante1.aprobado() else 'No'}")
27
28     print(f"--- Estudiante 2: {estudiante2.nombre} ---")
29     promedio2 = estudiante2.calcular_promedio()
30     print(f"Promedio: {promedio2}")
31     print(f"¿Aprobó?: {'Sí' if estudiante2.aprobado() else 'No'}")
32
33     print(f"--- Estudiante 3: {estudiante3.nombre} ---")
34     promedio3 = estudiante3.calcular_promedio()
35     print(f"Promedio: {promedio3}")
36     print(f"¿Aprobó?: {'Sí' if estudiante3.aprobado() else 'No'}")
```

CORRIDA:

```
PS C:\Users\LENOVO\Desktop\Programacion 11> &
--- Resultados de los estudiantes ---
--- Estudiante 1: Joel ---
Promedio: 7.75
¿Aprobó?: Sí
--- Estudiante 2: Carlos ---
Promedio: 5.75
¿Aprobó?: No
--- Estudiante 3: Bray ---
Promedio: 9.25
¿Aprobó?: Sí
PS C:\Users\LENOVO\Desktop\Programacion 11>
```

7 CODIGO EN JAVA:

CLASE CELULAR:

```
ma de Inicio  X  Salida - práctica de programación II (run)  X  PrácticaDeProgramaciónII.java  X  Celular.java  X
File History View Insert Tools Window Help
package practica.de.programacion.ll;

import java.util.ArrayList;
import java.util.List;

public class Celular {
    private static final int MAX_APPLICACIONES = 20;
    private static final int ESPACIO_MAXIMO_MB = 1024;
    private List<String> aplicacionesInstaladas;
    private int espacioOcupadoMB;
    private int bateriaPorcentaje;
    private boolean encendido;

    public Celular() {
        this.aplicacionesInstaladas = new ArrayList<>();
        this.espacioOcupadoMB = 0;
        this.bateriaPorcentaje = 100;
        this.encendido = true;
    }

    // a) Crea un método para instalar una nueva aplicación
    public boolean instalarAplicacion(String nombreAplicacion, int pesoMB) {
        if (!encendido) {
            System.out.println("El celular está apagado. No se pueden instalar aplicaciones.");
            return false;
        }
        if (aplicacionesInstaladas.size() < MAX_APPLICACIONES && (espacioOcupadoMB + pesoMB) <= ESPACIO_MAXIMO_MB) {
            aplicacionesInstaladas.add(nombreAplicacion);
            espacioOcupadoMB += pesoMB;
            System.out.println("Aplicación " + nombreAplicacion + " instalada exitosamente. Espacio ocupado: " + espacioOcupadoMB + "MB.");
            return true;
        } else {
            System.out.println("No se pudo instalar " + nombreAplicacion + ". Límite de aplicaciones o espacio alcanzado.");
            return false;
        }
    }

    // b) Crea un método para utilizar una aplicación
    public void utilizarAplicacion(String nombreAplicacion, int tiempoUsoMinutos) {
        if (!encendido) {
            System.out.println("El celular está apagado. No se pueden utilizar aplicaciones.");
            return;
        }
        if (aplicacionesInstaladas.contains(nombreAplicacion)) {
            int minutosDeUso = tiempoUsoMinutos;
            while (minutosDeUso > 0 && bateriaPorcentaje > 0) {
                int tiempoASimular = Math.min(minutosDeUso, 10);
                bateriaPorcentaje -= (int) ((tiempoASimular * 100) / 1000);
                minutosDeUso -= 10;
            }
            System.out.println("Aplicación " + nombreAplicacion + " utilizada por " + tiempoUsoMinutos + " minutos. Batería restante: " + bateriaPorcentaje + "%");
        } else {
            System.out.println("La aplicación " + nombreAplicacion + " no está instalada en el celular.");
        }
    }
}
```

PRINCIPAL:

```
package practica.de.programacion.ll;

public class PracticaDeProgramacionLl {

    public static void main(String[] args) {
        Celular miCelular = new Celular();

        miCelular.instalarAplicacion("WhatsApp", 30);
        miCelular.instalarAplicacion("Instagram", 80);
        miCelular.instalarAplicacion("JuegoLiviano", 60);
        miCelular.instalarAplicacion("EditorVideo", 120);
        miCelular.instalarAplicacion("JuegoPesado", 350);
        miCelular.instalarAplicacion("OtraApp", 20);
        // Intentar instalar más aplicaciones para probar el límite
        for (int i = 7; i <= 25; i++) {
            miCelular.instalarAplicacion("App" + i, 10);
        }

        miCelular.mostrarBateria();

        miCelular.utilizarAplicacion("WhatsApp", 25);
        miCelular.utilizarAplicacion("JuegoLiviano", 60);
        miCelular.utilizarAplicacion("EditorVideo", 30);
        miCelular.utilizarAplicacion("JuegoPesado", 40);
        miCelular.utilizarAplicacion("Instagram", 90);
        miCelular.utilizarAplicacion("OtraApp", 120);

        miCelular.mostrarBateria();

        // Intentar usar el celular cuando la batería se agotó (si sucedió)
        miCelular.instalarAplicacion("NuevaApp", 15);
        miCelular.utilizarAplicacion("WhatsApp", 10);
    }
}
```

CORRIDA:

```
Página de Inicio x Salida - práctica de programación II (run) x PrácticaDeProgramaciónLljava x Celular.java x
run:
Aplicación 'WhatsApp' instalada exitosamente. Espacio ocupado: 30MB.
Aplicación 'Instagram' instalada exitosamente. Espacio ocupado: 110MB.
Aplicación 'JuegoLíviano' instalada exitosamente. Espacio ocupado: 170MB.
Aplicación 'EditorVideo' instalada exitosamente. Espacio ocupado: 290MB.
Aplicación 'JuegoPesado' instalada exitosamente. Espacio ocupado: 640MB.
Aplicación 'OtraApp' instalada exitosamente. Espacio ocupado: 660MB.
Aplicación 'App7' instalada exitosamente. Espacio ocupado: 670MB.
Aplicación 'App8' instalada exitosamente. Espacio ocupado: 680MB.
Aplicación 'App9' instalada exitosamente. Espacio ocupado: 690MB.
Aplicación 'App10' instalada exitosamente. Espacio ocupado: 700MB.
Aplicación 'App11' instalada exitosamente. Espacio ocupado: 710MB.
Aplicación 'App12' instalada exitosamente. Espacio ocupado: 720MB.
Aplicación 'App13' instalada exitosamente. Espacio ocupado: 730MB.
Aplicación 'App14' instalada exitosamente. Espacio ocupado: 740MB.
Aplicación 'App15' instalada exitosamente. Espacio ocupado: 750MB.
Aplicación 'App16' instalada exitosamente. Espacio ocupado: 760MB.
Aplicación 'App17' instalada exitosamente. Espacio ocupado: 770MB.
Aplicación 'App18' instalada exitosamente. Espacio ocupado: 780MB.
Aplicación 'App19' instalada exitosamente. Espacio ocupado: 790MB.
Aplicación 'App20' instalada exitosamente. Espacio ocupado: 800MB.
No se pudo instalar 'App21'. Límite de aplicaciones o espacio alcanzado.
No se pudo instalar 'App22'. Límite de aplicaciones o espacio alcanzado.
No se pudo instalar 'App23'. Límite de aplicaciones o espacio alcanzado.
No se pudo instalar 'App24'. Límite de aplicaciones o espacio alcanzado.
No se pudo instalar 'App25'. Límite de aplicaciones o espacio alcanzado.
Porcentaje de batería restante: 100%.
Utilizando 'WhatsApp' por 10 minutos. Batería restante: 99%.
Utilizando 'WhatsApp' por 10 minutos. Batería restante: 98%.
Utilizando 'WhatsApp' por 5 minutos. Batería restante: 97%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 96%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 95%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 94%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 93%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 92%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 91%.
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 89%.
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 87%.
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 85%.
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 80%.
Página de Inicio x Salida - práctica de programación II (run) x PrácticaDeProgramaciónLljava x Celular.java x
Utilizando 'WhatsApp' por 5 minutos. Batería restante: 97%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 96%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 95%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 94%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 93%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 92%.
Utilizando 'JuegoLíviano' por 10 minutos. Batería restante: 91%.
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 89%.
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 87%.
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 85%.
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 80%.
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 75%.
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 70%.
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 65%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 64%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 63%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 62%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 61%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 60%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 59%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 58%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 57%.
Utilizando 'Instagram' por 10 minutos. Batería restante: 56%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 55%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 54%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 53%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 52%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 51%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 50%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 49%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 48%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 47%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 46%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 45%.
Utilizando 'OtraApp' por 10 minutos. Batería restante: 44%.
Porcentaje de batería restante: 44%.
No se pudo instalar 'NuevaApp'. Límite de aplicaciones o espacio alcanzado.
Utilizando 'WhatsApp' por 10 minutos. Batería restante: 43%.
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

CLASE CELULAR:

```
celular.py -> celular > __init__.py
```

```
class Celular:
    MAX_APPLICACIONES = 20
    ESPACIO_MAXIMO_MB = 1024

    def __init__(self):
        self.aplicaciones = []
        self.espacio_ocupado = 0
        self.bateria = 100
        self.encendido = True

    # a) Crea un método para instalar una nueva aplicación
    def instalar_aplicacion(self, nombre, peso_mb):
        if not self.encendido:
            print("El celular está apagado. No se pueden instalar aplicaciones.")
            return False
        if len(self.aplicaciones) < self.MAX_APPLICACIONES and (self.espacio_ocupado + peso_mb) <= self.ESPACIO_MAXIMO_MB:
            self.aplicaciones.append(nombre)
            self.espacio_ocupado += peso_mb
            print(f"Aplicación '{nombre}' instalada exitosamente. Espacio ocupado: {self.espacio_ocupado}MB.")
            return True
        else:
            print(f"No se pudo instalar '{nombre}'. Límite de aplicaciones o espacio alcanzado.")
            return False

    # b) Crea un método para utilizar una aplicación
    def utilizar_aplicacion(self, nombre_app, tiempo_uso_minutos):
        if not self.encendido:
            print("El celular está apagado. No se pueden utilizar aplicaciones.")
            return
        if nombre_app in self.aplicaciones:
            peso_mb = self.aplicaciones[nombre_app]
            consumo_por_10_minutos = 0
            if peso_mb > 250:
                consumo_por_10_minutos = 5
            elif peso_mb > 100:
                consumo_por_10_minutos = 2
            else:
                consumo_por_10_minutos = 1

            tiempo_restante = tiempo_uso_minutos
            while tiempo_restante > 0 and self.bateria > 0:
                uso = min(tiempo_restante, 10)
                self.bateria -= consumo_por_10_minutos * (uso / 10)
                tiempo_restante -= uso
                print(f"Utilizando '{nombre_app}' por {uso} minutos. Batería restante: {self.bateria:.2f}%")
                if self.bateria <= 0:
                    self.bateria = 0
                    self.encendido = False
                    print("¡Batería agotada! Celular apagado.")
                    return
            else:
                print(f"La aplicación '{nombre_app}' no está instalada.")

    # c) Muestra el porcentaje de batería restante
    def mostrar_bateria(self):
        print(f"Porcentaje de batería restante: {self.bateria:.2f}%")

    # d) Cuando la batería se acabe al tratar de utilizar el celular este debe mostrar el
    # mensaje de celular apagado (esto ya se maneja en utilizar_aplicacion)

# Ejemplo de uso
mi_celular = Celular()

mi_celular.instalar_aplicacion("WhatsApp", 30)
mi_celular.instalar_aplicacion("Instagram", 80)
mi_celular.instalar_aplicacion("JuegoLiviano", 60)
mi_celular.instalar_aplicacion("EditorVideo", 120)
mi_celular.instalar_aplicacion("JuegoPesado", 350)
mi_celular.instalar_aplicacion("OtraApp", 20)
# Intentar instalar más aplicaciones para probar los límites
for i in range(7, 26):
    mi_celular.instalar_aplicacion(f"App{i}", 10)

mi_celular.mostrar_bateria()

mi_celular.utilizar_aplicacion("WhatsApp", 25)
mi_celular.utilizar_aplicacion("JuegoLiviano", 60)
mi_celular.utilizar_aplicacion("EditorVideo", 30)
mi_celular.utilizar_aplicacion("JuegoPesado", 40)
mi_celular.utilizar_aplicacion("Instagram", 90)
mi_celular.utilizar_aplicacion("OtraApp", 120)

mi_celular.mostrar_bateria()

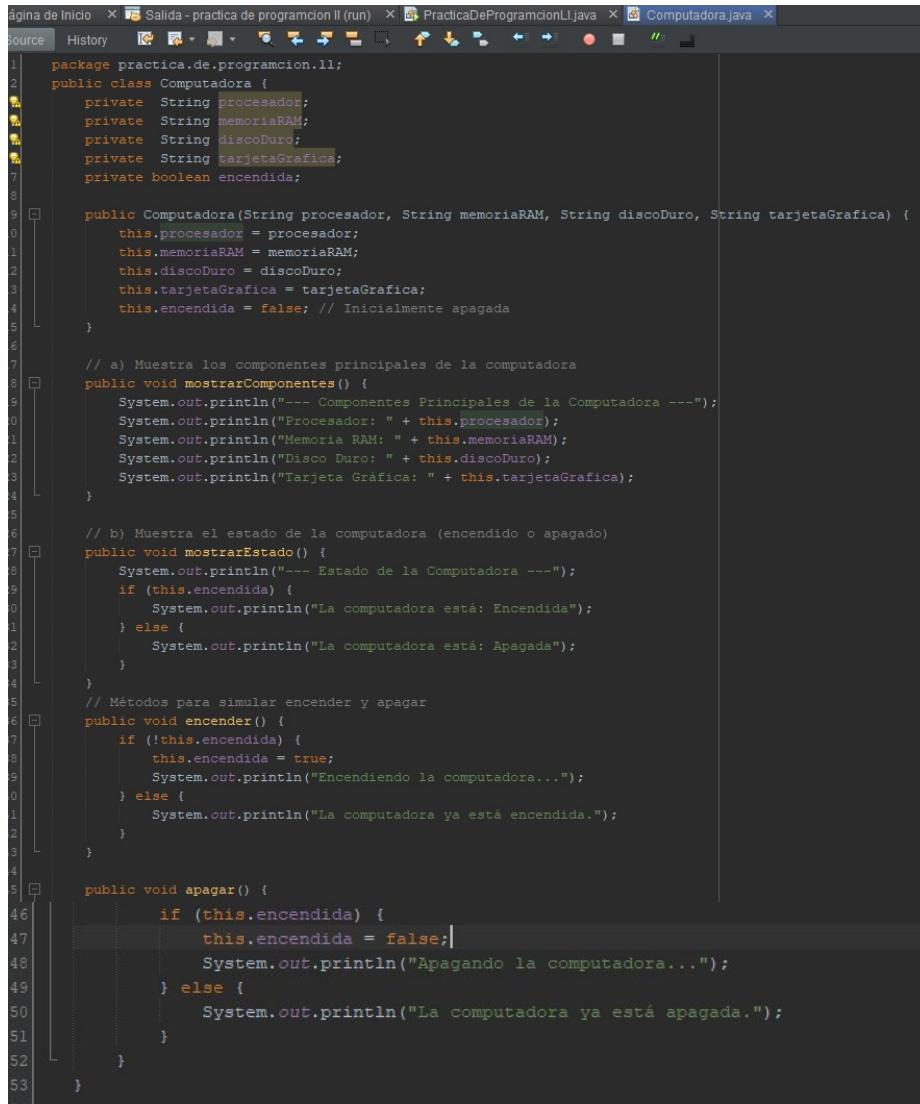
# Intentar usar el celular cuando la batería se agotó (si sucedió)
mi_celular.instalar_aplicacion("NuevaApp", 15)
mi_celular.utilizar_aplicacion("WhatsApp", 10)
```

CORRIDA:

```
PS C:\Users\LENOVO\Desktop\Programacion 11> & C:/Users/LENOVO/AppData/Local/  
Aplicación 'WhatsApp' instalada exitosamente. Espacio ocupado: 30MB.  
Aplicación 'Instagram' instalada exitosamente. Espacio ocupado: 110MB.  
Aplicación 'JuegoLiviano' instalada exitosamente. Espacio ocupado: 170MB.  
Aplicación 'EditorVideo' instalada exitosamente. Espacio ocupado: 290MB.  
Aplicación 'JuegoPesado' instalada exitosamente. Espacio ocupado: 640MB.  
Aplicación 'OtraApp' instalada exitosamente. Espacio ocupado: 660MB.  
Aplicación 'App7' instalada exitosamente. Espacio ocupado: 670MB.  
Aplicación 'App8' instalada exitosamente. Espacio ocupado: 680MB.  
Aplicación 'App9' instalada exitosamente. Espacio ocupado: 690MB.  
Aplicación 'App10' instalada exitosamente. Espacio ocupado: 700MB.  
Aplicación 'App11' instalada exitosamente. Espacio ocupado: 710MB.  
Aplicación 'App12' instalada exitosamente. Espacio ocupado: 720MB.  
Aplicación 'App13' instalada exitosamente. Espacio ocupado: 730MB.  
Aplicación 'App14' instalada exitosamente. Espacio ocupado: 740MB.  
Aplicación 'App15' instalada exitosamente. Espacio ocupado: 750MB.  
Aplicación 'App16' instalada exitosamente. Espacio ocupado: 760MB.  
Aplicación 'App17' instalada exitosamente. Espacio ocupado: 770MB.  
Aplicación 'App18' instalada exitosamente. Espacio ocupado: 780MB.  
Aplicación 'App19' instalada exitosamente. Espacio ocupado: 790MB.  
Aplicación 'App20' instalada exitosamente. Espacio ocupado: 800MB.  
No se pudo instalar 'App21'. Límite de aplicaciones o espacio alcanzado.  
No se pudo instalar 'App22'. Límite de aplicaciones o espacio alcanzado.  
No se pudo instalar 'App23'. Límite de aplicaciones o espacio alcanzado.  
No se pudo instalar 'App24'. Límite de aplicaciones o espacio alcanzado.  
No se pudo instalar 'App25'. Límite de aplicaciones o espacio alcanzado.  
Porcentaje de batería restante: 100.00%.  
Utilizando 'WhatsApp' por 10 minutos. Batería restante: 99.00%.  
Utilizando 'WhatsApp' por 10 minutos. Batería restante: 98.00%.  
Utilizando 'WhatsApp' por 5 minutos. Batería restante: 97.50%.  
Utilizando 'JuegoLiviano' por 10 minutos. Batería restante: 96.50%.  
Utilizando 'JuegoLiviano' por 10 minutos. Batería restante: 95.50%.  
Utilizando 'JuegoLiviano' por 10 minutos. Batería restante: 94.50%.  
Utilizando 'JuegoLiviano' por 10 minutos. Batería restante: 93.50%.  
Utilizando 'JuegoLiviano' por 10 minutos. Batería restante: 92.50%.  
Utilizando 'JuegoLiviano' por 10 minutos. Batería restante: 91.50%.  
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 89.50%.  
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 87.50%.  
Utilizando 'EditorVideo' por 10 minutos. Batería restante: 85.50%.  
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 80.50%.  
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 75.50%.  
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 75.50%.  
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 70.50%.  
Utilizando 'JuegoPesado' por 10 minutos. Batería restante: 65.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 64.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 63.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 62.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 61.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 60.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 59.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 58.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 57.50%.  
Utilizando 'Instagram' por 10 minutos. Batería restante: 56.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 55.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 54.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 53.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 52.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 51.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 50.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 49.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 48.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 47.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 46.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 45.50%.  
Utilizando 'OtraApp' por 10 minutos. Batería restante: 44.50%.  
Porcentaje de batería restante: 44.50%.  
No se pudo instalar 'NuevaApp'. Límite de aplicaciones o espacio alcanzado.  
Utilizando 'WhatsApp' por 10 minutos. Batería restante: 43.50%.
```

9 CODIGO EN JAVA:

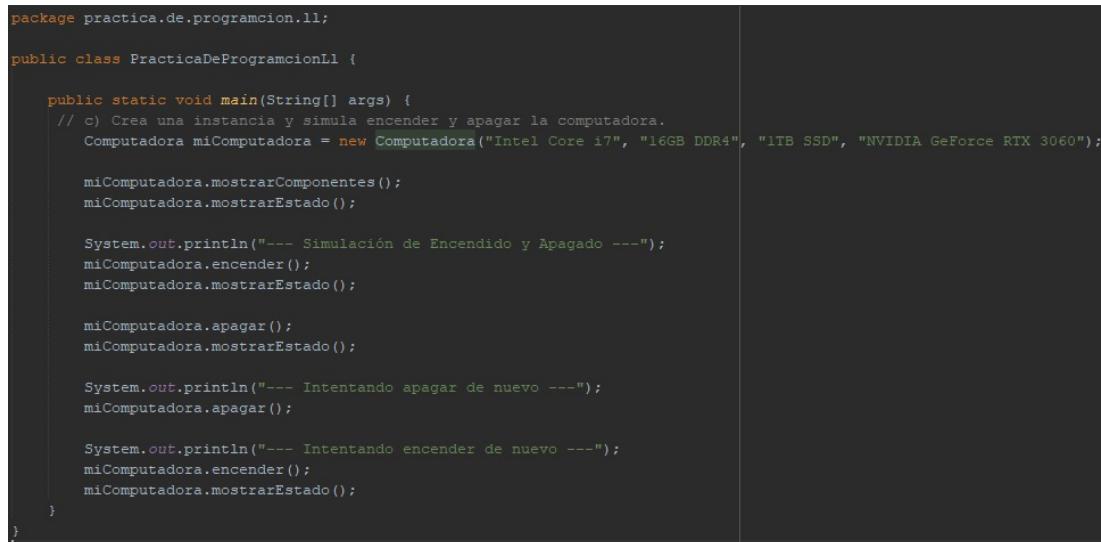
CALSE COMPUTADORA:



The screenshot shows a Java code editor with the following code:

```
1 package practica.de.programacion.ll;
2 public class Computadora {
3     private String procesador;
4     private String memoriaRAM;
5     private String discoDuro;
6     private String tarjetaGrafica;
7     private boolean encendida;
8
9     public Computadora(String procesador, String memoriaRAM, String discoDuro, String tarjetaGrafica) {
10         this.procesador = procesador;
11         this.memoriaRAM = memoriaRAM;
12         this.discoDuro = discoDuro;
13         this.tarjetaGrafica = tarjetaGrafica;
14         this.encendida = false; // Inicialmente apagada
15     }
16
17     // a) Muestra los componentes principales de la computadora
18     public void mostrarComponentes() {
19         System.out.println("--- Componentes Principales de la Computadora ---");
20         System.out.println("Procesador: " + this.procesador);
21         System.out.println("Memoria RAM: " + this.memoriaRAM);
22         System.out.println("Disco Duro: " + this.discoDuro);
23         System.out.println("Tarjeta Gráfica: " + this.tarjetaGrafica);
24     }
25
26     // b) Muestra el estado de la computadora (encendido o apagado)
27     public void mostrarEstado() {
28         System.out.println("--- Estado de la Computadora ---");
29         if (this.encendida) {
30             System.out.println("La computadora está: Encendida");
31         } else {
32             System.out.println("La computadora está: Apagada");
33         }
34     }
35
36     // Métodos para simular encender y apagar
37     public void encender() {
38         if (!this.encendida) {
39             this.encendida = true;
40             System.out.println("Encendiendo la computadora...");
41         } else {
42             System.out.println("La computadora ya está encendida.");
43         }
44     }
45
46     public void apagar() {
47         if (this.encendida) {
48             this.encendida = false;
49             System.out.println("Apagando la computadora...");
50         } else {
51             System.out.println("La computadora ya está apagada.");
52         }
53     }
54 }
```

PRINCIPAL:



The screenshot shows a Java code editor with the following code:

```
package practica.de.programacion.ll;

public class PracticaDeProgramacionLl {

    public static void main(String[] args) {
        // c) Crea una instancia y simula encender y apagar la computadora.
        Computadora miComputadora = new Computadora("Intel Core i7", "16GB DDR4", "1TB SSD", "NVIDIA GeForce RTX 3060");

        miComputadora.mostrarComponentes();
        miComputadora.mostrarEstado();

        System.out.println("--- Simulación de Encendido y Apagado ---");
        miComputadora.encender();
        miComputadora.mostrarEstado();

        miComputadora.apagar();
        miComputadora.mostrarEstado();

        System.out.println("--- Intentando apagar de nuevo ---");
        miComputadora.apagar();

        System.out.println("--- Intentando encender de nuevo ---");
        miComputadora.encender();
        miComputadora.mostrarEstado();
    }
}
```

CORRIDA:

```
run:  
--- Componentes Principales de la Computadora ---  
Procesador: Intel Core i7  
Memoria RAM: 16GB DDR4  
Disco Duro: 1TB SSD  
Tarjeta Gráfica: NVIDIA GeForce RTX 3060  
--- Estado de la Computadora ---  
La computadora está: Apagada  
--- Simulación de Encendido y Apagado ---  
Encendiendo la computadora...  
--- Estado de la Computadora ---  
La computadora está: Encendida  
Apagando la computadora...  
--- Estado de la Computadora ---  
La computadora está: Apagada  
--- Intentando apagar de nuevo ---  
La computadora ya está apagada.  
--- Intentando encender de nuevo ---  
Encendiendo la computadora...  
--- Estado de la Computadora ---  
La computadora está: Encendida  
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

CLASE COMPUTADRA:

CORRIDA:

```
ejercicio5.py > ⚙ Computadora > ⚡ apagar
1 class Computadora:
2     def __init__(self, procesador, memoria_ram, disco_duro, tarjeta_grafica):
3         self.procesador = procesador
4         self.memoria_ram = memoria_ram
5         self.disco_duro = disco_duro
6         self.tarjeta_grafica = tarjeta_grafica
7         self.encendida = False # Inicialmente apagada
8
9     # a) Muestra Los componentes principales de La computadora
10    def mostrar_componentes(self):
11        print("--- Componentes Principales de la Computadora ---")
12        print(f"Procesador: {self.procesador}")
13        print(f"Memoria RAM: {self.memoria_ram}")
14        print(f"Disco Duro: {self.disco_duro}")
15        print(f"Tarjeta Gráfica: {self.tarjeta_grafica}")
16
17    # b) Muestra el estado de la computadora (encendido o apagado)
18    def mostrar_estado(self):
19        print("--- Estado de la Computadora ---")
20        if self.encendida:
21            print("La computadora está: Encendida")
22        else:
23            print("La computadora está: Apagada")
24
25    # Métodos para simular encender y apagar
26    def encender(self):
27        if not self.encendida:
28            self.encendida = True
29            print("Encendiendo la computadora...")
30        else:
31            print("La computadora ya está encendida.")
32
33    def apagar(self):
34        if self.encendida:
35            self.encendida = False
36            print("Apagando la computadora...")
37        else:
38            print("La computadora ya está apagada.")
```

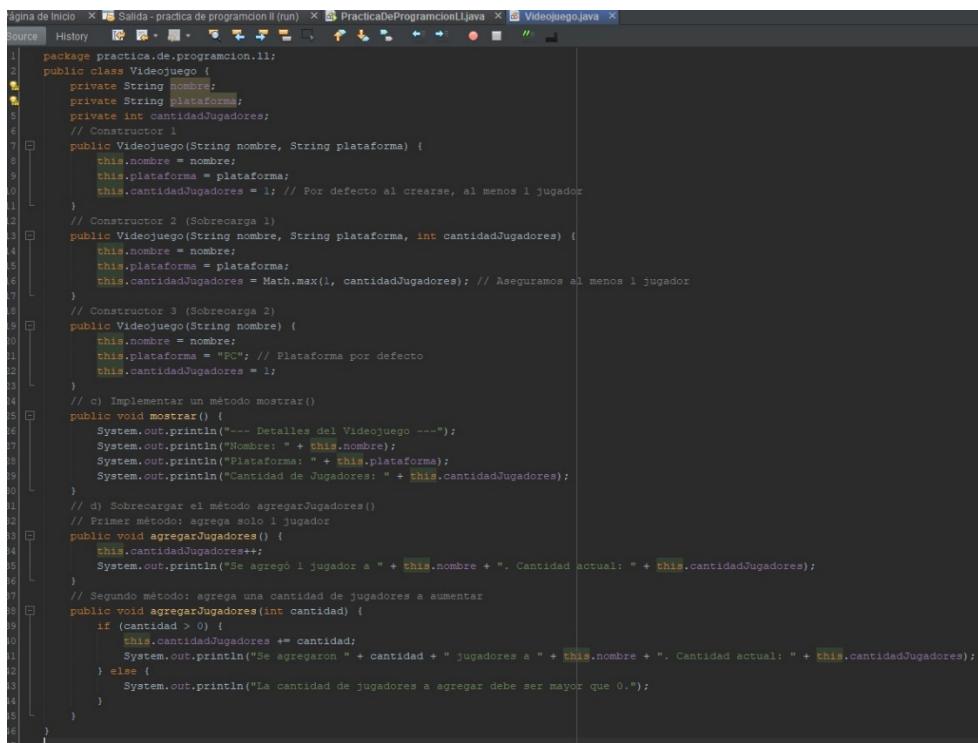
```
ejercicio5.py > ⚙ Computadora > ⚡ apagar
1 class Computadora:
2     def encender(self):
26         if not self.encendida:
27             self.encendida = True
28             print("Encendiendo la computadora...")
29         else:
30             print("La computadora ya está encendida.")
32
33     def apagar(self):
34         if self.encendida:
35             self.encendida = False
36             print("Apagando la computadora...")
37         else:
38             print("La computadora ya está apagada.")
39
40     # c) Crea una instancia y simula encender y apagar la computadora.
41     mi_computadora = Computadora("Intel Core i7", "16GB DDR4", "1TB SSD", "NVIDIA GeForce RTX 3060")
42
43     mi_computadora.mostrar_componentes()
44     mi_computadora.mostrar_estado()
45
46     print("--- Simulación de Encendido y Apagado ---")
47     mi_computadora.encender()
48     mi_computadora.mostrar_estado()
49
50     mi_computadora.apagar()
51     mi_computadora.mostrar_estado()
52
53     print("--- Intentando apagar de nuevo ---")
54     mi_computadora.apagar()
55
56     print("--- Intentando encender de nuevo ---")
57     mi_computadora.encender()
58     mi_computadora.mostrar_estado()
```

```
--- Estado de la Computadora ---
La computadora está: Apagada
--- Simulación de Encendido y Apagado ---
Encendiendo la computadora...
--- Estado de la Computadora ---
La computadora está: Encendida
Apagando la computadora...
--- Estado de la Computadora ---
La computadora está: Apagada
--- Intentando apagar de nuevo ---
La computadora ya está apagada.
--- Intentando encender de nuevo ---
Encendiendo la computadora...
--- Estado de la Computadora ---
La computadora está: Encendida
PS C:\Users\LENOVO\Desktop\Programacion 11> 
```

POLIMORFISMO

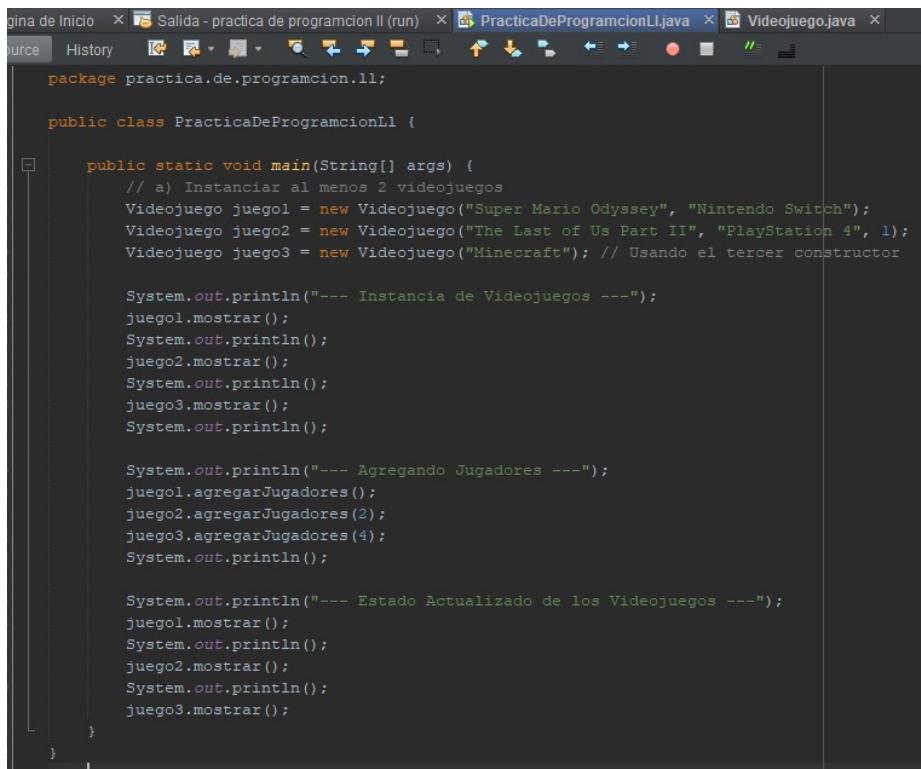
1 CODIGO EN JAVA:

CLASE VIDEOJUEGO:



```
package practica.de.programacion.ll;
public class Videojuego {
    private String nombre;
    private String plataforma;
    private int cantidadJugadores;
    // Constructor 1
    public Videojuego(String nombre, String plataforma) {
        this.nombre = nombre;
        this.plataforma = plataforma;
        this.cantidadJugadores = 1; // Por defecto al crearse, al menos 1 jugador
    }
    // Constructor 2 (Sobrecarga 1)
    public Videojuego(String nombre, String plataforma, int cantidadJugadores) {
        this.nombre = nombre;
        this.plataforma = plataforma;
        this.cantidadJugadores = Math.max(1, cantidadJugadores); // Aseguramos al menos 1 jugador
    }
    // Constructor 3 (Sobrecarga 2)
    public Videojuego(String nombre) {
        this.nombre = nombre;
        this.plataforma = "PC"; // Plataforma por defecto
        this.cantidadJugadores = 1;
    }
    // c) Implementar un método mostrar()
    public void mostrar() {
        System.out.println("--- Detalles del Videojuego ---");
        System.out.println("Nombre: " + this.nombre);
        System.out.println("Plataforma: " + this.plataforma);
        System.out.println("Cantidad de Jugadores: " + this.cantidadJugadores);
    }
    // d) Sobrecargar el método agregarJugadores()
    // Primer método: agrega solo 1 jugador
    public void agregarJugadores() {
        this.cantidadJugadores++;
        System.out.println("Se agregó 1 jugador a " + this.nombre + ". Cantidad actual: " + this.cantidadJugadores);
    }
    // Segundo método: agrega una cantidad de jugadores a aumentar
    public void agregarJugadores(int cantidad) {
        if (cantidad > 0) {
            this.cantidadJugadores += cantidad;
            System.out.println("Se agregaron " + cantidad + " jugadores a " + this.nombre + ". Cantidad actual: " + this.cantidadJugadores);
        } else {
            System.out.println("La cantidad de jugadores a agregar debe ser mayor que 0.");
        }
    }
}
```

PRINCIPAL:



```
package practica.de.programacion.ll;

public class PracticaDeProgramacionLl {

    public static void main(String[] args) {
        // a) Instanciar al menos 2 videojuegos
        Videojuego juego1 = new Videojuego("Super Mario Odyssey", "Nintendo Switch");
        Videojuego juego2 = new Videojuego("The Last of Us Part II", "PlayStation 4", 1);
        Videojuego juego3 = new Videojuego("Minecraft"); // Usando el tercer constructor

        System.out.println("--- Instancia de Videojuegos ---");
        juego1.mostrar();
        System.out.println();
        juego2.mostrar();
        System.out.println();
        juego3.mostrar();
        System.out.println();

        System.out.println("--- Agregando Jugadores ---");
        juego1.agregarJugadores();
        juego2.agregarJugadores(2);
        juego3.agregarJugadores(4);
        System.out.println();

        System.out.println("--- Estado Actualizado de los Videojuegos ---");
        juego1.mostrar();
        System.out.println();
        juego2.mostrar();
        System.out.println();
        juego3.mostrar();
    }
}
```

CORRIDA:

```
run:  
--- Instancia de Videojuegos ---  
--- Detalles del Videojuego ---  
Nombre: Super Mario Odyssey  
Plataforma: Nintendo Switch  
Cantidad de Jugadores: 1  
  
--- Detalles del Videojuego ---  
Nombre: The Last of Us Part II  
Plataforma: PlayStation 4  
Cantidad de Jugadores: 1  
  
--- Detalles del Videojuego ---  
Nombre: Minecraft  
Plataforma: PC  
Cantidad de Jugadores: 1  
  
--- Agregando Jugadores ---  
Se agregó 1 jugador a Super Mario Odyssey. Cantidad actual: 2  
Se agregaron 2 jugadores a The Last of Us Part II. Cantidad actual: 3  
Se agregaron 4 jugadores a Minecraft. Cantidad actual: 5  
  
--- Estado Actualizado de los Videojuegos ---  
--- Detalles del Videojuego ---  
Nombre: Super Mario Odyssey  
Plataforma: Nintendo Switch  
Cantidad de Jugadores: 2  
  
--- Detalles del Videojuego ---  
Nombre: The Last of Us Part II  
Plataforma: PlayStation 4  
Cantidad de Jugadores: 3  
  
--- Detalles del Videojuego ---  
Nombre: Minecraft  
Plataforma: PC  
Cantidad de Jugadores: 5  
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

CLASE VIDEOJUEGO:

```
ejercicio6.py > ...
1  #POLIMORFISMO
2  class Videojuego:
3      def __init__(self, nombre, plataforma="PC", cantidadJugadores=1):
4          """Constructor principal con valores por defecto."""
5          self.nombre = nombre
6          self.plataforma = plataforma
7          self.cantidadJugadores = max(1, cantidadJugadores)
8
9      # c) Implementar un método mostrar()
10     def mostrar(self):
11         print("--- Detalles del Videojuego ---")
12         print(f"Nombre: {self.nombre}")
13         print(f"Plataforma: {self.plataforma}")
14         print(f"Cantidad de Jugadores: {self.cantidadJugadores}")
15
16     # d) Sobrecargar el método agregarJugadores()
17     # Primer método: agrega solo 1 jugador
18     def agregarJugadores(self):
19         self.cantidadJugadores += 1
20         print(f"Se agregó 1 jugador a {self.nombre}. Cantidad actual: {self.cantidadJugadores}")
21
22     # Segundo método: agrega una cantidad de jugadores a aumentar
23     def agregarJugadores(self, cantidad):
24         if cantidad > 0:
25             self.cantidadJugadores += cantidad
26
27             print(f"Se agregaron {cantidad} jugadores a {self.nombre}. Cantidad actual: {self.cantidadJugadores}")
28         else:
29             print("La cantidad de jugadores a agregar debe ser mayor que 0.")
30
31     # a) Instanciar al menos 2 videojuegos
32     juego1 = Videojuego("Super Mario Odyssey", "Nintendo Switch")
33     juego2 = Videojuego("The Last of Us Part II", "PlayStation 4", 1)
34     juego3 = Videojuego("Minecraft") # Usando el constructor con solo nombre (aprovechando el valor por defecto)
35
36     print("---- Instancia de Videojuegos ----")
37     juego1.mostrar()
38     print()
39     juego2.mostrar()
40     print()
41     juego3.mostrar()
42     print()
43
44     print("---- Agregando Jugadores ----")
45     juego1.agregarJugadores()
46     juego2.agregarJugadores(2)
47     juego3.agregarJugadores(4)
48     print()
49
50     print("---- Estado Actualizado de los Videojuegos ----")
51     juego1.mostrar()
52     print()
53     juego2.mostrar()
54     print()
55     juego3.mostrar()
```

CORRIDA:

```
PS C:\Users\LENOVO\Desktop\Programacion 11> & C:/Users/LENOVO/AppData/Local/Microsoft/WindowsApp
--- Instancia de Videojuegos ---
--- Detalles del Videojuego ---
Nombre: Super Mario Odyssey
Plataforma: Nintendo Switch
Cantidad de Jugadores: 1

--- Detalles del Videojuego ---
Nombre: The Last of Us Part II
Plataforma: PlayStation 4
Cantidad de Jugadores: 1

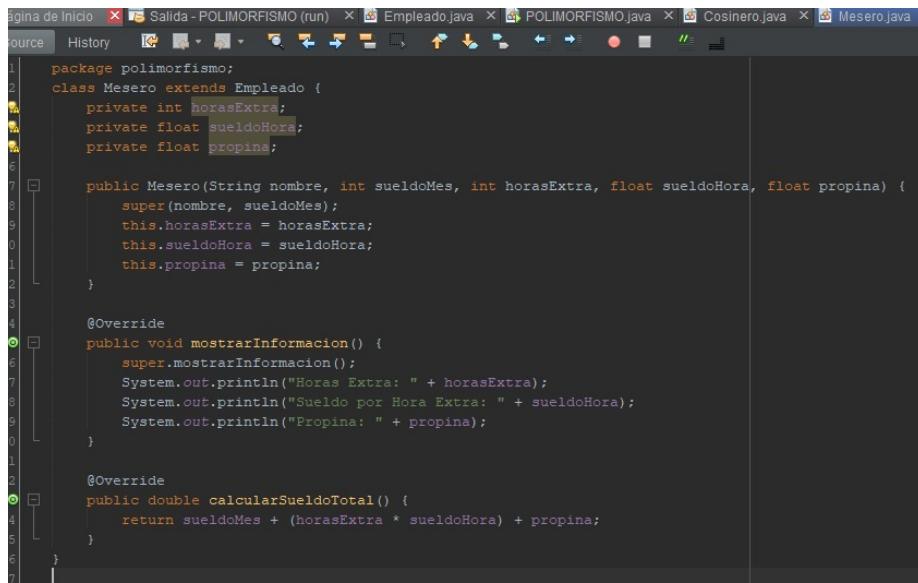
--- Detalles del Videojuego ---
Nombre: Minecraft
Plataforma: PC
Cantidad de Jugadores: 1

--- Agregando Jugadores ---
Traceback (most recent call last):
  File "c:/Users/LENOVO/Desktop/Programacion 11/ejercicio6.py", line 44, in <module>
    juego1.agregarJugadores()
TypeError: Videojuego.agregarJugadores() missing 1 required positional argument: 'cantidad'
PS C:\Users\LENOVO\Desktop\Programacion 11>
```

3 CODIGO EN JAVA:

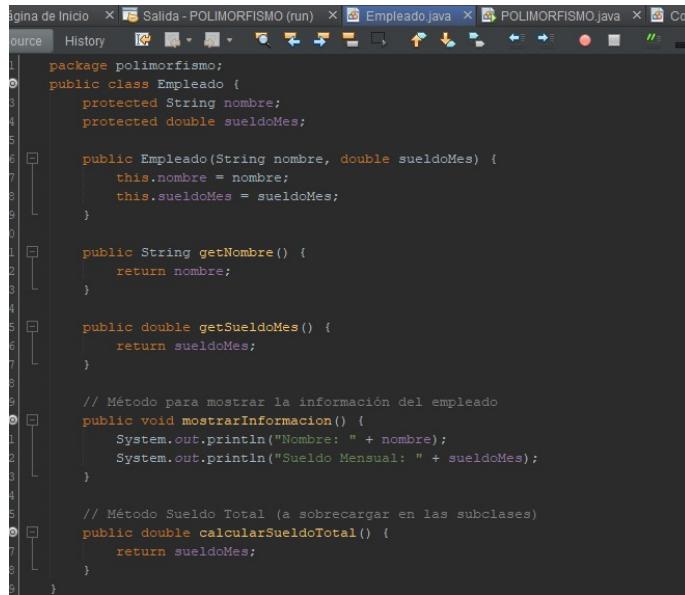
RESTAURANTE:

CLASE MESERO:



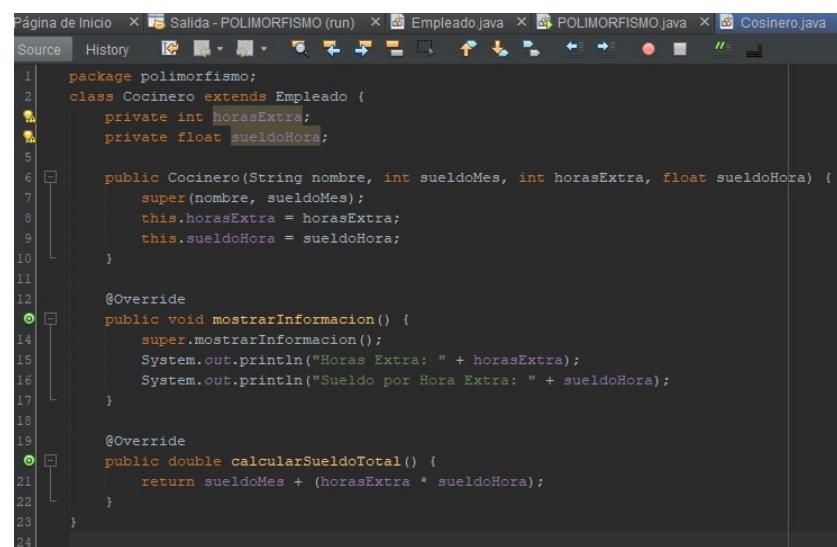
```
1 package polimorfismo;
2 class Mesero extends Empleado {
3     private int horasExtra;
4     private float sueldoHora;
5     private float propina;
6
7     public Mesero(String nombre, int sueldoMes, int horasExtra, float sueldoHora, float propina) {
8         super(nombre, sueldoMes);
9         this.horasExtra = horasExtra;
10        this.sueldoHora = sueldoHora;
11        this.propina = propina;
12    }
13
14    @Override
15    public void mostrarInformacion() {
16        super.mostrarInformacion();
17        System.out.println("Horas Extra: " + horasExtra);
18        System.out.println("Sueldo por Hora Extra: " + sueldoHora);
19        System.out.println("Propina: " + propina);
20    }
21
22    @Override
23    public double calcularSueldoTotal() {
24        return sueldoMes + (horasExtra * sueldoHora) + propina;
25    }
26}
```

CLASE EMPLEADO:



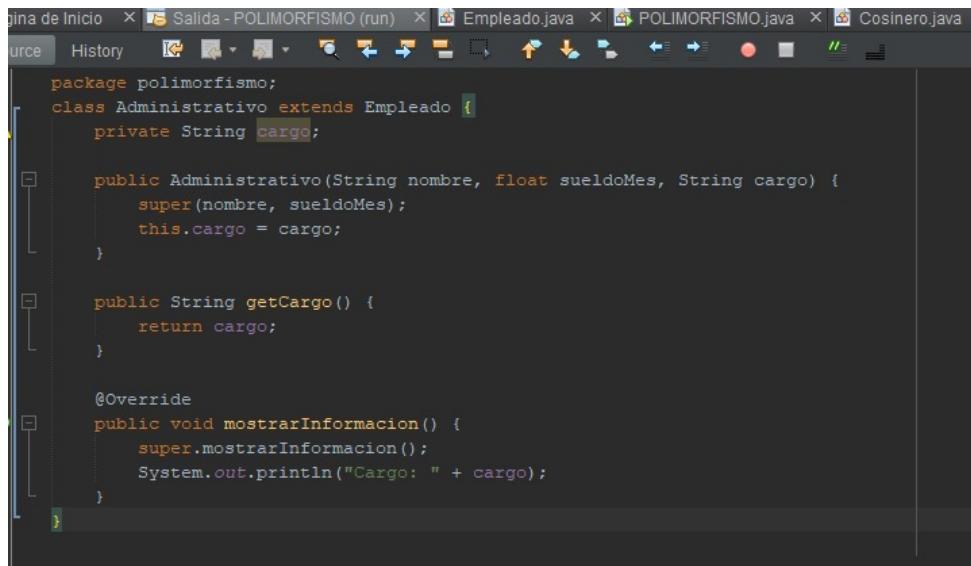
```
1 package polimorfismo;
2 public class Empleado {
3     protected String nombre;
4     protected double sueldoMes;
5
6     public Empleado(String nombre, double sueldoMes) {
7         this.nombre = nombre;
8         this.sueldoMes = sueldoMes;
9     }
10
11    public String getNombre() {
12        return nombre;
13    }
14
15    public double getSueldoMes() {
16        return sueldoMes;
17    }
18
19    // Método para mostrar la información del empleado
20    public void mostrarInformacion() {
21        System.out.println("Nombre: " + nombre);
22        System.out.println("Sueldo Mensual: " + sueldoMes);
23    }
24
25    // Método Sueldo Total (a sobrecargar en las subclases)
26    public double calcularSueldoTotal() {
27        return sueldoMes;
28    }
29}
```

CLASE COSINERO:



```
1 package polimorfismo;
2 class Cocinero extends Empleado {
3     private int horasExtra;
4     private float sueldoHora;
5
6     public Cocinero(String nombre, int sueldoMes, int horasExtra, float sueldoHora) {
7         super(nombre, sueldoMes);
8         this.horasExtra = horasExtra;
9         this.sueldoHora = sueldoHora;
10    }
11
12    @Override
13    public void mostrarInformacion() {
14        super.mostrarInformacion();
15        System.out.println("Horas Extra: " + horasExtra);
16        System.out.println("Sueldo por Hora Extra: " + sueldoHora);
17    }
18
19    @Override
20    public double calcularSueldoTotal() {
21        return sueldoMes + (horasExtra * sueldoHora);
22    }
23}
```

CLASE ADMINISTRATIVO:



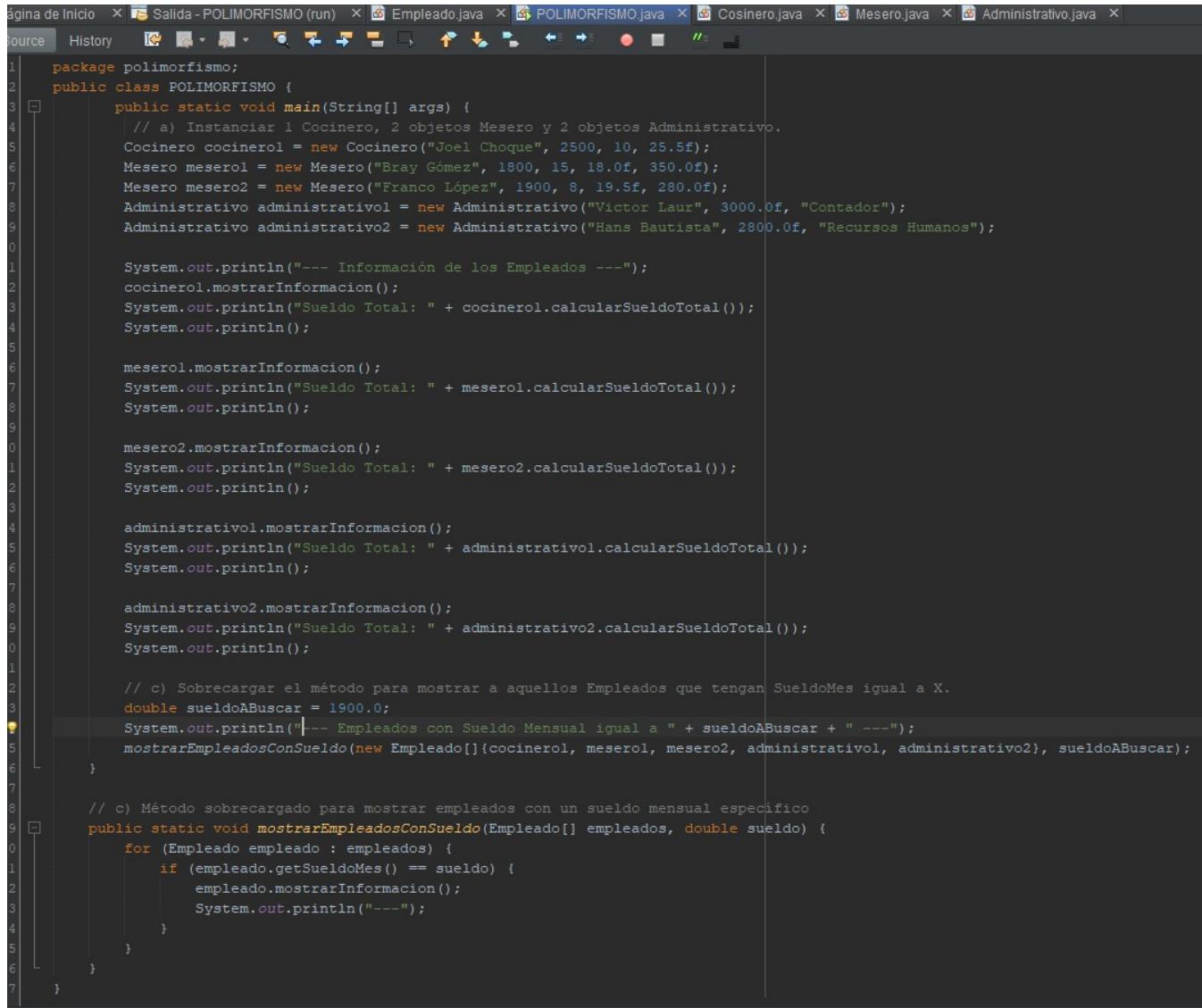
```
package polimorfismo;
class Administrativo extends Empleado {
    private String cargo;

    public Administrativo(String nombre, float sueldoMes, String cargo) {
        super(nombre, sueldoMes);
        this.cargo = cargo;
    }

    public String getCargo() {
        return cargo;
    }

    @Override
    public void mostrarInformacion() {
        super.mostrarInformacion();
        System.out.println("Cargo: " + cargo);
    }
}
```

PRINCIPAL:



```
package polimorfismo;
public class POLIMORFISMO {
    public static void main(String[] args) {
        // a) Instanciar 1 Cocinero, 2 objetos Mesero y 2 objetos Administrativo.
        Cocinero cocinerol = new Cocinero("Joel Choque", 2500, 10, 25.5f);
        Mesero meserol = new Mesero("Bray Gómez", 1800, 15, 18.0f, 350.0f);
        Mesero mesero2 = new Mesero("Franco López", 1900, 8, 19.5f, 280.0f);
        Administrativo administrativol = new Administrativo("Victor Laur", 3000.0f, "Contador");
        Administrativo administrativo2 = new Administrativo("Hans Bautista", 2800.0f, "Recursos Humanos");

        System.out.println("--- Información de los Empleados ---");
        cocinerol.mostrarInformacion();
        System.out.println("Sueldo Total: " + cocinerol.calcularSueldoTotal());
        System.out.println();

        meserol.mostrarInformacion();
        System.out.println("Sueldo Total: " + meserol.calcularSueldoTotal());
        System.out.println();

        mesero2.mostrarInformacion();
        System.out.println("Sueldo Total: " + mesero2.calcularSueldoTotal());
        System.out.println();

        administrativol.mostrarInformacion();
        System.out.println("Sueldo Total: " + administrativol.calcularSueldoTotal());
        System.out.println();

        administrativo2.mostrarInformacion();
        System.out.println("Sueldo Total: " + administrativo2.calcularSueldoTotal());
        System.out.println();

        // c) Sobrecargar el método para mostrar a aquellos Empleados que tengan SueldoMes igual a X.
        double sueldoABuscar = 1900.0;
        System.out.println("--- Empleados con Sueldo Mensual igual a " + sueldoABuscar + " ---");
        mostrarEmpleadosConSueldo(new Empleado[]{cocinerol, meserol, mesero2, administrativol, administrativo2}, sueldoABuscar);
    }

    // c) Método sobrecargado para mostrar empleados con un sueldo mensual específico
    public static void mostrarEmpleadosConSueldo(Empleado[] empleados, double sueldo) {
        for (Empleado empleado : empleados) {
            if (empleado.getSueldoMes() == sueldo) {
                empleado.mostrarInformacion();
                System.out.println("----");
            }
        }
    }
}
```

CORRIDA:

```
Página de Inicio  ×  Salida - POLIMORFISMO (run)  ×  Empleado.java  ×  POLIMORFISMO.java
--- Información de los Empleados ---
Nombre: Joel Choque
Sueldo Mensual: 2500.0
Horas Extra: 10
Sueldo por Hora Extra: 25.5
Sueldo Total: 2755.0

Nombre: Bray Gómez
Sueldo Mensual: 1800.0
Horas Extra: 15
Sueldo por Hora Extra: 18.0
Propina: 350.0
Sueldo Total: 2420.0

Nombre: Franco López
Sueldo Mensual: 1900.0
Horas Extra: 8
Sueldo por Hora Extra: 19.5
Propina: 280.0
Sueldo Total: 2336.0

Nombre: Victor Laur
Sueldo Mensual: 3000.0
Cargo: Contador
Sueldo Total: 3000.0

Nombre: Hans Bautista
Sueldo Mensual: 2800.0
Cargo: Recursos Humanos
Sueldo Total: 2800.0

--- Empleados con Sueldo Mensual igual a 1900.0 ---
Nombre: Franco López
Sueldo Mensual: 1900.0
Horas Extra: 8
Sueldo por Hora Extra: 19.5
Propina: 280.0
---
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

RESTAURANTE:

```
ejercicio7.py > ...
1  class Empleado:
2      def __init__(self, nombre, sueldo_mes):
3          self.nombre = nombre
4          self.sueldo_mes = sueldo_mes
5
6      def mostrar_informacion(self):
7          print(f"Nombre: {self.nombre}")
8          print(f"Sueldo Mensual: {self.sueldo_mes}")
9
10     def calcular_sueldo_total(self):
11         return self.sueldo_mes
12
13 class Cocinero(Empleado):
14     def __init__(self, nombre, sueldo_mes, horas_extra, sueldo_hora):
15         super().__init__(nombre, sueldo_mes)
16         self.horas_extra = horas_extra
17         self.sueldo_hora = sueldo_hora
18
19     def mostrar_informacion(self):
20         super().mostrar_informacion()
21         print(f"Horas Extra: {self.horas_extra}")
22         print(f"Sueldo por Hora Extra: {self.sueldo_hora}")
23
24     def calcular_sueldo_total(self):
25         return self.sueldo_mes + (self.horas_extra * self.sueldo_hora)
26
27 class Mesero(Empleado):
28     def __init__(self, nombre, sueldo_mes, horas_extra, sueldo_hora, propina):
29         super().__init__(nombre, sueldo_mes)
30         self.horas_extra = horas_extra
31         self.sueldo_hora = sueldo_hora
32         self.propina = propina
33
34     def mostrar_informacion(self):
35         super().mostrar_informacion()
36         print(f"Horas Extra: {self.horas_extra}")
37         print(f"Sueldo por Hora Extra: {self.sueldo_hora}")
```

```
ejercicio7.py > ...
27  class Mesero(Empleado):
28      def mostrar_informacion(self):
29          super().mostrar_informacion()
30          print(f"Horas Extra: {self.horas_extra}")
31          print(f"Sueldo por Hora Extra: {self.sueldo_hora}")
32          print(f"Propina: {self.propina}")
33
34      def calcular_sueldo_total(self):
35          return self.sueldo_mes + (self.horas_extra * self.sueldo_hora) + self.propina
36
37  class Administrativo(Empleado):
38      def __init__(self, nombre, sueldo_mes, cargo):
39          super().__init__(nombre, sueldo_mes)
40          self.cargo = cargo
41
42      def mostrar_informacion(self):
43          super().mostrar_informacion()
44          print(f"Cargo: {self.cargo}")
45
46 # a) Instanciar 1 Cocinero, 2 objetos Mesero y 2 objetos Administrativo.
47 cocinero1 = Cocinero("Joel Choque", 2500, 10, 25.5)
48 mesero1 = Mesero("Bray Gómez", 1800, 15, 18.0, 350.0)
49 mesero2 = Mesero("Franco López", 1900, 8, 19.5, 280.0)
50 administrativo1 = Administrativo("Víctor Laura", 3000.0, "Contador")
51 administrativo2 = Administrativo("Hans Bautista", 2800.0, "Recursos Humanos")
52
53 print("--- Información de los Empleados ---")
54 cocinero1.mostrar_informacion()
55 print(f"Sueldo Total: {cocinero1.calcular_sueldo_total()}")
56 print()
57
58 mesero1.mostrar_informacion()
59 print(f"Sueldo Total: {mesero1.calcular_sueldo_total()}")
60 print()
61
62 mesero2.mostrar_informacion()
63 print(f"Sueldo Total: {mesero2.calcular_sueldo_total()}")
64
```

```

ejercicio7.py > ...
  2     print()
  3
  4     mesero1.mostrar_informacion()
  5     print(f"Sueldo Total: {mesero1.calcular_sueldo_total()}")
  6     print()
  7
  8     mesero2.mostrar_informacion()
  9     print(f"Sueldo Total: {mesero2.calcular_sueldo_total()}")
 10    print()
 11
 12    administrativo1.mostrar_informacion()
 13    print(f"Sueldo Total: {administrativo1.calcular_sueldo_total()}")
 14    print()
 15
 16    administrativo2.mostrar_informacion()
 17    print(f"Sueldo Total: {administrativo2.calcular_sueldo_total()}")
 18    print()
 19
 20    # c) Sobre cargar el método para mostrar a aquellos Empleados que tengan SueldoMes igual a X.
 21    def mostrar_empleados_con_sueldo(empleados, sueldo_objetivo):
 22        print(f"\n--- Empleados con Sueldo Mensual igual a {sueldo_objetivo} ---")
 23        for empleado in empleados:
 24            if empleado.sueldo_mes == sueldo_objetivo:
 25                empleado.mostrar_informacion()
 26                print("----")
 27
 28    empleados_restaurante = [cocinero1, mesero1, mesero2, administrativo1, administrativo2]
 29    sueldo_a_buscar = 1900
 30    mostrar_empleados_con_sueldo(empleados_restaurante, sueldo_a_buscar)

```

CORRIDA:

```

PS C:\Users\LENOVO\Desktop\Programacion 11> & C:/Use
--- Información de los Empleados ---
Nombre: Joel Choque
Sueldo Mensual: 2500
Horas Extra: 10
Sueldo por Hora Extra: 25.5
Sueldo Total: 2755.0

Nombre: Bray Gómez
Sueldo Mensual: 1800
Horas Extra: 15
Sueldo por Hora Extra: 18.0
Propina: 350.0
Sueldo Total: 2420.0

Nombre: Franco López
Sueldo Mensual: 1900
Horas Extra: 8
Sueldo por Hora Extra: 19.5
Propina: 280.0
Sueldo Total: 2336.0

Nombre: Victor Laura
Sueldo Mensual: 3000.0
Cargo: Contador
Sueldo Total: 3000.0

Nombre: Hans Bautista
Sueldo Mensual: 2800.0
Cargo: Recursos Humanos
Sueldo Total: 2800.0

--- Empleados con Sueldo Mensual igual a 1900 ---
Nombre: Franco López
Sueldo Mensual: 1900
Horas Extra: 8
Sueldo por Hora Extra: 19.5
Propina: 280.0
---
```

5 CODIGO EN JAVA:

AMBIENTES DE LA UNIVERSIDAD:

CLASE AULA:

The screenshot shows a Java code editor with the following code:

```
1 package polimorfismo;
2 class Aula extends Ambiente {
3     private int capacidad;
4     private int nropupitres;
5
6     public Aula(String nombre, int capacidad, int nropupitres) {
7         super(nombre);
8         this.capacidad = capacidad;
9         this.nropupitres = nropupitres;
10    }
11
12    @Override
13    public void mostrar() {
14        super.mostrar();
15        System.out.println("Capacidad: " + capacidad);
16        System.out.println("Número de pupitres: " + nropupitres);
17    }
18
19    @Override
20    public int cantidadMuebles() {
21        return nropupitres; // En un aula, los pupitres son los muebles principales
22    }
23}
```

CLASE OFICINA:

The screenshot shows a Java code editor with the following code:

```
1 package polimorfismo;
2 class Oficina extends Ambiente {
3     private int nroSillas;
4     private int nroEscritorios;
5     private int nroEstanterias;
6
7     public Oficina(String nombre, int nroSillas, int nroEscritorios, int nroEstanterias) {
8         super(nombre);
9         this.nroSillas = nroSillas;
10        this.nroEscritorios = nroEscritorios;
11        this.nroEstanterias = nroEstanterias;
12    }
13
14    @Override
15    public void mostrar() {
16        super.mostrar();
17        System.out.println("Número de sillas: " + nroSillas);
18        System.out.println("Número de escritorios: " + nroEscritorios);
19        System.out.println("Número de estanterías: " + nroEstanterias);
20    }
21
22    @Override
23    public int cantidadMuebles() {
24        return nroSillas + nroEscritorios + nroEstanterias;
25    }
26}
```

CLASE LABORATORIO:

The screenshot shows the Java code for the `Laboratorio` class within an IDE interface. The code defines a class that extends `Ambiente`, with fields for capacity, number of tables, and number of chairs. It overrides the `mostrar()` and `cantidadMuebles()` methods.

```
1 package polimorfismo;
2 class Laboratorio extends Ambiente {
3     private int capacidad;
4     private int nroMesas;
5     private int nroSillas;
6
7     public Laboratorio(String nombre, int capacidad, int nroMesas, int nroSillas) {
8         super(nombre);
9         this.capacidad = capacidad;
10        this.nroMesas = nroMesas;
11        this.nroSillas = nroSillas;
12    }
13
14    @Override
15    public void mostrar() {
16        super.mostrar();
17        System.out.println("Capacidad: " + capacidad);
18        System.out.println("Número de mesas: " + nroMesas);
19        System.out.println("Número de sillas: " + nroSillas);
20    }
21
22    @Override
23    public int cantidadMuebles() {
24        return nroMesas + nroSillas;
25    }
26}
27
```

CLASE AMBIENTE:

The screenshot shows the Java code for the `Ambiente` class within an IDE interface. It is a base class with a protected `nombre` field, a constructor, and methods to get the name and print it, as well as a method to calculate the number of furniture pieces.

```
1 package polimorfismo;
2 public class Ambiente {
3     protected String nombre;
4
5     public Ambiente(String nombre) {
6         this.nombre = nombre;
7     }
8
9     public String getNombre() {
10        return nombre;
11    }
12
13    public void mostrar() {
14        System.out.println("Nombre del ambiente: " + nombre);
15    }
16
17    public int cantidadMuebles() {
18        return 0; // Método base, se sobrecargará en las subclases
19    }
20}
```

CLASE PRINCIPAL:

```
1 package polimorfismo;
2 public class POLIMORFISMO {
3     public static void main(String[] args) {
4         // a) Instanciar 2 objetos Oficina, 2 Aulas y 1 Laboratorio
5         Oficina oficina = new Oficina("Oficina de Dirección de ", 3, 2, 4);
6         Oficina oficina2 = new Oficina("Kardex", 2, 3, 3);
7
8         Aula aula1 = new Aula("Aula P1-A1", 50, 50);
9         Aula aula2 = new Aula("Aula Sub Suelo", 35, 35);
10
11         Laboratorio laboratoriol = new Laboratorio("Laboratorio de Programación", 20, 8, 16);
12
13         // b) Crear un método mostrar() para mostrar los datos de cada objeto
14         System.out.println("--- Datos de las Oficinas ---");
15         oficina.mostrar();
16         System.out.println("---");
17         oficina2.mostrar();
18         System.out.println("---");
19
20         System.out.println("--- Datos de las Aulas ---");
21         aula1.mostrar();
22         System.out.println("---");
23         aula2.mostrar();
24         System.out.println("---");
25
26         System.out.println("--- Datos del Laboratorio ---");
27         laboratoriol.mostrar();
28         System.out.println("---");
29
30         // c) Sobrecargar el método cantidadMuebles()
31         System.out.println("--- Cantidad de Muebles por Ambiente ---");
32         System.out.println(oficina.getNombre() + ": " + oficina.cantidadMuebles() + " muebles");
33         System.out.println(oficina2.getNombre() + ": " + oficina2.cantidadMuebles() + " muebles");
34         System.out.println(aula1.getNombre() + ": " + aula1.cantidadMuebles() + " muebles");
35         System.out.println(aula2.getNombre() + ": " + aula2.cantidadMuebles() + " muebles");
36         System.out.println(laboratoriol.getNombre() + ": " + laboratoriol.cantidadMuebles() + " muebles");
37     }
38 }
```

CORRIDA:

```
Página de Inicio × Salida - POLIMORFISMO (run) × POLIMORFISMO.java × Laboratorio.j
▶ run:
--- Datos de las Oficinas ---
Nombre del ambiente: Oficina de Dirección de
Número de sillas: 3
Número de escritorios: 2
Número de estanterías: 4
---
Nombre del ambiente: Kardex
Número de sillas: 2
Número de escritorios: 3
Número de estanterías: 3
---
--- Datos de las Aulas ---
Nombre del ambiente: Aula P1-A1
Capacidad: 50
Número de pupitres: 50
---
Nombre del ambiente: Aula Sub Suelo
Capacidad: 35
Número de pupitres: 35
---
--- Datos del Laboratorio ---
Nombre del ambiente: Laboratorio de Programación
Capacidad: 20
Número de mesas: 8
Número de sillas: 16
---
--- Cantidad de Muebles por Ambiente ---
Oficina de Dirección de : 9 muebles
Kardex: 8 muebles
Aula P1-A1: 50 muebles
Aula Sub Suelo: 35 muebles
Laboratorio de Programación: 24 muebles
BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON

UNIVERSIDAD:

```
ejercicio8.py > ...
1  class Ambiente:
2      def __init__(self, nombre):
3          self.nombre = nombre
4
5      def mostrar(self):
6          print(f"Nombre del ambiente: {self.nombre}")
7
8      def cantidad_muebles(self):
9          return 0 # Método base, se sobrecargará en las subclases
10
11 class Oficina(Ambiente):
12     def __init__(self, nombre, nro_sillas, nro_escritorios, nro_estanterias):
13         super().__init__(nombre)
14         self.nro_sillas = nro_sillas
15         self.nro_escritorios = nro_escritorios
16         self.nro_estanterias = nro_estanterias
17
18     def mostrar(self):
19         super().mostrar()
20         print(f"Número de sillas: {self.nro_sillas}")
21         print(f"Número de escritorios: {self.nro_escritorios}")
22         print(f"Número de estanterías: {self.nro_estanterias}")
23
24     def cantidad_muebles(self):
25         return self.nro_sillas + self.nro_escritorios + self.nro_estanterias
26
27 class Aula(Ambiente):
28     def __init__(self, nombre, capacidad, nro_pupitres):
29         super().__init__(nombre)
30         self.capacidad = capacidad
31         self.nro_pupitres = nro_pupitres
32
33     def mostrar(self):
34         super().mostrar()
35         print(f"Capacidad: {self.capacidad}")
36         print(f"Número de pupitres: {self.nro_pupitres}")
37
```

```
ejercicio8.py > ...
27  class Aula(Ambiente):
28      def mostrar(self):
29          print(f"Capacidad: {self.capacidad}")
30          print(f"Número de pupitres: {self.nro_pupitres}")
31
32      def cantidad_muebles(self):
33          return self.nro_pupitres # Consideramos los pupitres como los muebles principales
34
35  class Laboratorio(Ambiente):
36      def __init__(self, nombre, capacidad, nro_mesas, nro_sillas):
37          super().__init__(nombre)
38          self.capacidad = capacidad
39          self.nro_mesas = nro_mesas
40          self.nro_sillas = nro_sillas
41
42      def mostrar(self):
43          super().mostrar()
44          print(f"Capacidad: {self.capacidad}")
45          print(f"Número de mesas: {self.nro_mesas}")
46          print(f"Número de sillas: {self.nro_sillas}")
47
48      def cantidad_muebles(self):
49          return self.nro_mesas + self.nro_sillas
50
51 # a) Instanciar 2 objetos Oficina, 2 Aulas y 1 Laboratorio
52 oficina1 = Oficina("Oficina de Dirección de Carrera", 3, 2, 4)
53 oficina2 = Oficina("Kardex", 2, 3, 3)
54
55 aula1 = Aula("Aula P1-A1", 50, 50)
56 aula2 = Aula("Aula Sub Suelo", 35, 35)
57
58 laboratorio1 = Laboratorio("Laboratorio de Programación II", 20, 8, 16)
59
60 # b) Crear un método mostrar() para mostrar los datos de cada objeto
61 print("--- Datos de las Oficinas ---")
62 oficina1.mostrar()
63 print("---")
```

ejercicio8.py > ...

```
63
64     laboratorio1 = Laboratorio("Laboratorio de Programacion II", 20, 8, 16)
65
66     # b) Crear un método mostrar() para mostrar los datos de cada objeto
67     print("--- Datos de las Oficinas ---")
68     oficina1.mostrar()
69     print("---")
70     oficina2.mostrar()
71     print("---")
72
73     print("--- Datos de las Aulas ---")
74     aula1.mostrar()
75     print("---")
76     aula2.mostrar()
77     print("---")
78
79     print("--- Datos del Laboratorio ---")
80     laboratorio1.mostrar()
81     print("---")
82
83     # c) Sobrecargar el método cantidadMuebles()
84     print("--- Cantidad de Muebles por Ambiente ---")
85     print(f"{oficina1.nombre}: {oficina1.cantidad_muebles()} muebles")
86     print(f"{oficina2.nombre}: {oficina2.cantidad_muebles()} muebles")
87     print(f"{aula1.nombre}: {aula1.cantidad_muebles()} muebles")
88     print(f"{aula2.nombre}: {aula2.cantidad_muebles()} muebles")
89     print(f"{laboratorio1.nombre}: {laboratorio1.cantidad_muebles()} muebles")
```

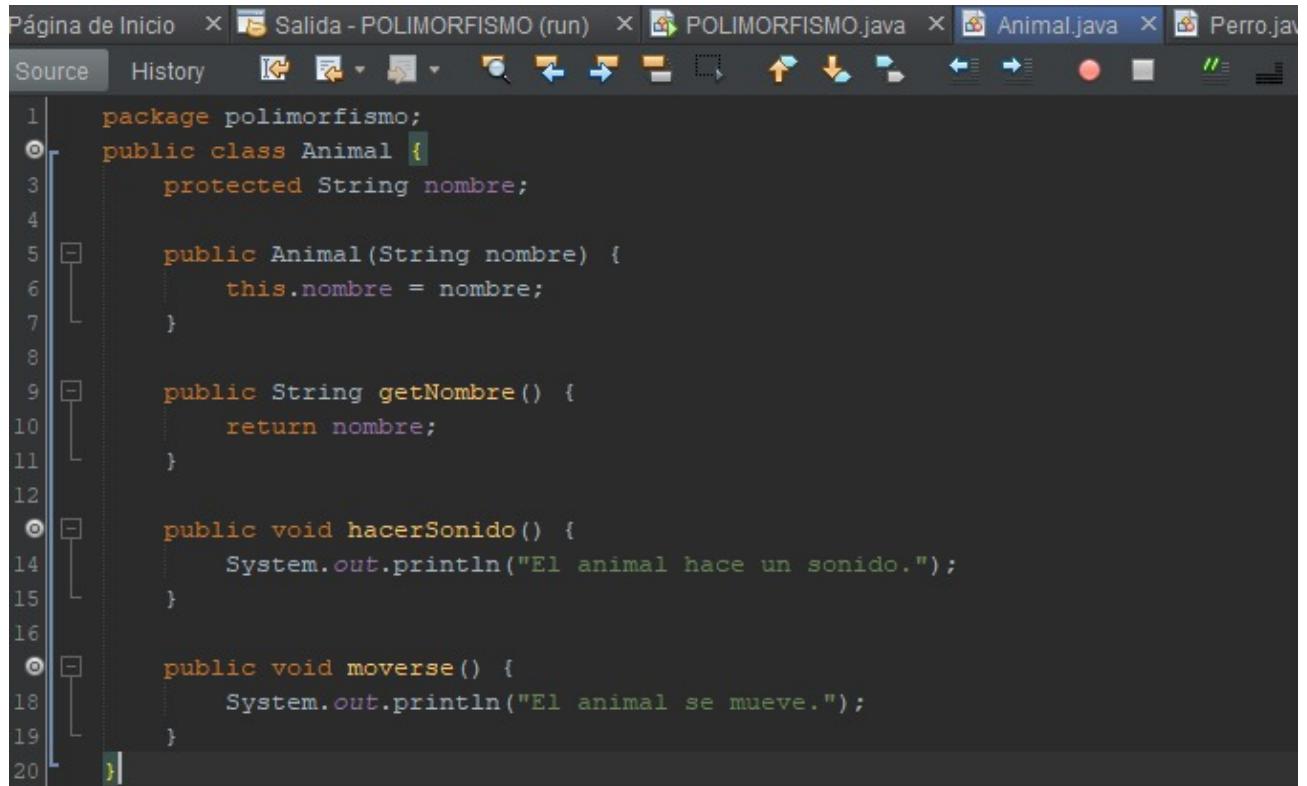
CORRIDA:

```
Capacidad: 35
Número de pupitres: 35
---
--- Datos del Laboratorio ---
Nombre del ambiente: Laboratorio de Programacion II
Capacidad: 20
Número de mesas: 8
Número de sillas: 16
---
--- Cantidad de Muebles por Ambiente ---
Oficina de Dirección de Carrera: 9 muebles
Kardex: 8 muebles
Aula P1-A1: 50 muebles
Aula Sub Suelo: 35 muebles
Laboratorio de Programacion II: 24 muebles
PS C:\Users\LENOVO\Desktop\Programacion 11>
```

9 CODIGO EN JAVA

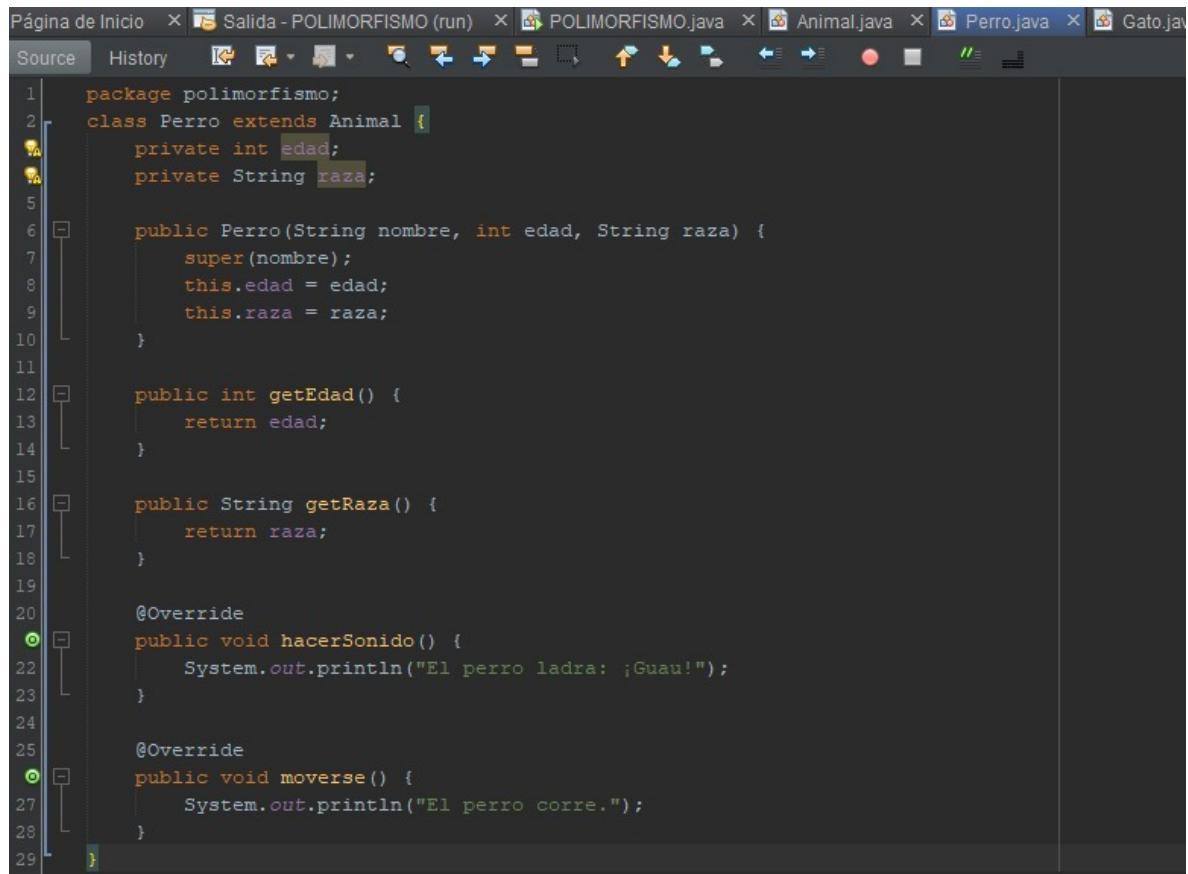
ANIMALES

CLASE ANIMAL:



```
1 package polimorfismo;
2 public class Animal {
3     protected String nombre;
4
5     public Animal(String nombre) {
6         this.nombre = nombre;
7     }
8
9     public String getNombre() {
10        return nombre;
11    }
12
13    public void hacerSonido() {
14        System.out.println("El animal hace un sonido.");
15    }
16
17    public void moverse() {
18        System.out.println("El animal se mueve.");
19    }
20}
```

CLASE PERRO:



```
1 package polimorfismo;
2 class Perro extends Animal {
3     private int edad;
4     private String raza;
5
6     public Perro(String nombre, int edad, String raza) {
7         super(nombre);
8         this.edad = edad;
9         this.raza = raza;
10    }
11
12    public int getEdad() {
13        return edad;
14    }
15
16    public String getRaza() {
17        return raza;
18    }
19
20    @Override
21    public void hacerSonido() {
22        System.out.println("El perro ladra: ¡Guau!");
23    }
24
25    @Override
26    public void moverse() {
27        System.out.println("El perro corre.");
28    }
29}
```

CLASE GATO:

```
1 package polimorfismo;
2 class Gato extends Animal {
3     private String color;
4
5     public Gato(String nombre, String color) {
6         super(nombre);
7         this.color = color;
8     }
9
10    public String getColor() {
11        return color;
12    }
13
14    @Override
15    public void hacerSonido() {
16        System.out.println("El gato maulla: ;Miau!");
17    }
18
19    @Override
20    public void moverse() {
21        System.out.println("El gato salta.");
22    }
23}
```

CLASE PAJARO:

```
1 package polimorfismo;
2 class Pajaro extends Animal {
3     private String tipo;
4
5     public Pajaro(String nombre, String tipo) {
6         super(nombre);
7         this.tipo = tipo;
8     }
9
10    public String getTipo() {
11        return tipo;
12    }
13
14    @Override
15    public void hacerSonido() {
16        System.out.println("El pájaro canta: ;Pio pio!");
17    }
18
19    @Override
20    public void moverse() {
21        System.out.println("El pájaro vuela.");
22    }
23}
```

CLASE PRINCIPAL:

```
1 package polimorfismo;
2 public class POLIMORFISMO {
3     public static void main(String[] args) {
4         // a) Instanciar 1 Perro, 1 Gato y 1 Pájaro.
5         Perro miPerro = new Perro("Firulais", 3, "Golden Retriever");
6         Gato miGato = new Gato("Pelusa", "Blanco");
7         Pajaro miPajaro = new Pajaro("Piolin", "Canario");
8
9         System.out.println("--- Información de los Animales ---");
10        System.out.println("Perro: Nombre=" + miPerro.getNombre() + ", Edad=" + miPerro.getEdad() + ", Raza=" + miPerro.getRaza());
11        System.out.println("Gato: Nombre=" + miGato.getNombre() + ", Color=" + miGato.getColor());
12        System.out.println("Pájaro: Nombre=" + miPajaro.getNombre() + ", Tipo=" + miPajaro.getTipo());
13        System.out.println();
14
15        // b) Sobrecargar el método hacerSonido() para que cada animal emita su sonido característico.
16        System.out.println("--- Sonidos de los Animales ---");
17        miPerro.hacerSonido();
18        miGato.hacerSonido();
19        miPajaro.hacerSonido();
20        System.out.println();
21
22        // c) Implementar un método moverse() que indique cómo se mueve cada animal (correr, saltar, volar, etc.).
23        System.out.println("--- Movimiento de los Animales ---");
24        miPerro.moverse();
25        miGato.moverse();
26        miPajaro.moverse();
27    }
28}
```

CORRIDA:

```
run:
--- Información de los Animales ---
Perro: Nombre=Firulais, Edad=3, Raza=Golden Retriever
Gato: Nombre=Pelusa, Color=Blanco
Pájaro: Nombre=Piolín, Tipo=Canario

--- Sonidos de los Animales ---
El perro ladra: ¡Guau!
El gato maúlla: ¡Miau!
El pájaro canta: ¡Pío pío!

--- Movimiento de los Animales ---
El perro corre.
El gato salta.
El pájaro vuela.

BUILD SUCCESSFUL (total time: 0 seconds)
```

CODIGO EN PYTHON:

CLASE ANIMAL:

```
ejercicio9.py > ...
1  class Animal:
2      def __init__(self, nombre):
3          self.nombre = nombre
4
5      def hacer_sonido(self):
6          print("El animal hace un sonido.")
7
8      def moverse(self):
9          print("El animal se mueve.")
10
11 class Perro(Animal):
12     def __init__(self, nombre, edad, raza):
13         super().__init__(nombre)
14         self.edad = edad
15         self.raza = raza
16
17     def hacer_sonido(self):
18         print("¡Guau!")
19
20     def moverse(self):
21         print("El perro corre.")
22
23 class Gato(Animal):
24     def __init__(self, nombre, color):
25         super().__init__(nombre)
26         self.color = color
27
28     def hacer_sonido(self):
29         print("¡Miau!")
30
31     def moverse(self):
32         print("El gato salta.")
33
34 class Pajaro(Animal):
35     def __init__(self, nombre, tipo):
36         super().__init__(nombre)
37         self.tipo = tipo
```

```

ejercicio9.py > ...
34  class Pajaro(Animal):
35      def __init__(self, nombre, tipo):
36          super().__init__(nombre)
37          self.tipo = tipo
38
39      def hacer_sonido(self):
40          print("¡Pío pío!")
41
42      def moverse(self):
43          print("El pájaro vuela.")
44
45 # a) Instanciar 1 Perro, 1 Gato y 1 Pájaro.
46 mi_perro = Perro("Firulais", 3, "Golden Retriever")
47 mi_gato = Gato("Pelusa", "Blanco")
48 mi_pajaro = Pajaro("Piolín", "Canario")
49
50 print("--- Información de los Animales ---")
51 print(f"Perro: Nombre={mi_perro.nombre}, Edad={mi_perro.edad}, Raza={mi_perro.raza}")
52 print(f"Gato: Nombre={mi_gato.nombre}, Color={mi_gato.color}")
53 print(f"Pájaro: Nombre={mi_pajaro.nombre}, Tipo={mi_pajaro.tipo}")
54 print()
55
56 # b) Sobreponer el método hacer_sonido() para que cada animal emita su sonido característico.
57 print("--- Sonidos de los Animales ---")
58 mi_perro.hacer_sonido()
59 mi_gato.hacer_sonido()
60 mi_pajaro.hacer_sonido()
61 print()
62
63 # c) Implementar un método moverse() que indique cómo se mueve cada animal (correr, saltar, volar, etc.).
64 print("--- Movimiento de los Animales ---")
65 mi_perro.moverse()
66 mi_gato.moverse()
67 mi_pajaro.moverse()

```

CORRIDA:

PROBLEMAS	SALIDA	CONSOLA DE DEPURACIÓN	<u>TERMINAL</u>	PUERTOS
			PS C:\Users\LENOVO\Desktop\Programacion 11> & C:/Users/LENOVO/Desktop/Programacion 11/ejercicio9.py	
			--- Información de los Animales ---	
			Perro: Nombre=Firulais, Edad=3, Raza=Golden Retriever	
			Gato: Nombre=Pelusa, Color=Blanco	
			Pájaro: Nombre=Piolín, Tipo=Canario	
			--- Sonidos de los Animales ---	
			¡Guau!	
			¡Miau!	
			¡Pío pío!	
			--- Movimiento de los Animales ---	
			El perro corre.	
			El gato salta.	
			El pájaro vuela.	
			PS C:\Users\LENOVO\Desktop\Programacion 11>	