# Resolución Práctico 7: Herencia y polimorfismo.

Ejercicio 1: En este ejercicio debemos crear una clase padre llamada Vehiculo con los atributos marca y modelo, y el método mostrarInfo(), luego debemos crear una clase Auto que herede de esta clase vehículo y además crearle un atributo adicional cantidadDePuertas, también debemos sobreescribir el método mostrarInfo de la clase padre. Por ultimo debemos instanciar un auto en el main y mostrar su información completa.

Codigo.  
Clase Vehiculo:  
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Clase Auto:  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Main:

Texto

El contenido generado por IA puede ser incorrecto.

Salida por pantalla:

Forma

El contenido generado por IA puede ser incorrecto.

Ejercicio 2:  
En este ejercicio debemos crear una clase abstracta llamada Figura con un atributo nombre y un método calcularArea(). Luego debemos crear dos clases que hereden de Figura: una clase Circulo y una clase Rectangulo, cada una implementando su propia versión del método calcularArea() de acuerdo a su fórmula correspondiente.  
Por último, debemos crear un array de figuras, instanciar al menos un círculo y un rectángulo, y mostrar el área de cada figura usando polimorfismo.

Codigo.  
Clase Figura:

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Clase Circulo:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Clase rectángulo:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Main:

Texto

El contenido generado por IA puede ser incorrecto.

Salida por pantalla:

Texto

El contenido generado por IA puede ser incorrecto.

Ejercicio 3:  
En este ejercicio debemos crear una clase Empleado con un método calcularSueldo(Empleado e) que reciba un empleado y devuelva un sueldo fijo según el tipo de empleado. También debemos crear dos clases que hereden de Empleado: EmpleadoPlanta y EmpleadoTemporal.  
Luego, debemos crear una lista de empleados, agregar instancias de ambas subclases, recorrer la lista y para cada empleado invocar el método calcularSueldo() para obtener su sueldo, usando instanceof dentro del método para determinar el tipo de empleado y asignar el sueldo correspondiente.

Codigo.  
Empleado:  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

EmpleadoPlanta:  
Texto

El contenido generado por IA puede ser incorrecto.

EmpleadoTemporal:  
Texto

El contenido generado por IA puede ser incorrecto.

Main:  
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Salida por pantalla:  
Texto

El contenido generado por IA puede ser incorrecto.

Ejercicio 4:  
En este ejercicio debemos crear una clase llamada Animal con los métodos hacerSonido() y describirAnimal(). Luego, debemos crear tres clases que hereden de Animal: Perro, Gato y Vaca, y sobrescribir el método hacerSonido() en cada una con la anotación @Override para que cada animal emita su sonido característico.  
Por último, debemos crear una lista de animales, agregar instancias de Perro, Gato y Vaca, y mostrar los sonidos de cada uno utilizando polimorfismo.

Codigo.

Clase animal:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Clase Perro:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Clase Gato:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Clase Vaca:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Main:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Salida por consola:

Forma, Rectángulo

El contenido generado por IA puede ser incorrecto.

Ejercicio 5:  
En este ejercicio debemos crear una interfaz llamada Pagable con un método pagar(). Luego, debemos crear tres clases que implementen esta interfaz: TarjetaCredito, Transferencia y Efectivo, definiendo en cada una su propia versión del método pagar().  
Además, debemos crear un método llamado procesarPago(Pagable medio), que reciba cualquier forma de pago e invoque su método pagar(), permitiendo así procesar distintos medios de pago de forma genérica.  
Por último, debemos crear objetos de las distintas clases de pago y procesarlos usando una única función.

Codigo.  
Interfaz pagable:  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Clase TarjetaCredito:  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.  
Clase Transferencia:  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.  
Clase Efectivo:  
Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Main:  
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.  
Salida por consola:  
Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.