Universidad
Nacional de San Martín

Guía de Ejercicios

Unidad 21: Pilares de la POO

1 Ejercicios

1.1 Ejercicio 1: Vehículos y Herencia

Crea una clase base `Vehiculo` con atributos como `marca` y `modelo`. Luego, crea clases hijas `Auto`, `Motocicleta` y `Bicicleta` que hereden de `Vehiculo`. Cada clase hija debe implementar su propio método `arrancar`. Practica la herencia y el polimorfismo invocando el método `arrancar` en una lista de diferentes tipos de vehículos.

1.2 Ejercicio 2: Biblioteca de Libros

Crea una clase abstracta `Libro` con métodos abstractos `titulo()` y `autor()`. Luego, implementa clases hijas `LibroDigital` y `LibroFisico` que sobrescriban los métodos abstractos para mostrar el título y autor del libro. Practica la abstracción y la herencia.

1.3 Ejercicio 3: Sistema Bancario

Implementa una clase `CuentaBancaria` con atributos privados `saldo` y métodos públicos `depositar()` y `retirar()`. Luego, crea una clase hija `CuentaAhorros` que añada un atributo `interes`. Practica encapsulamiento al no permitir el acceso directo al saldo.

1.4 Ejercicio 4: Figuras Geométricas

Crea una clase abstracta `Figura` con el método abstracto `area()`. Implementa clases hijas `Cuadrado`, `Círculo` y `Triángulo`, cada una calculando su propia área. Practica abstracción y polimorfismo llamando al método `area()` en una lista de diferentes figuras.

1.5 Ejercicio 5: Sistema de Transporte

Diseña una clase base `Transporte` que contenga atributos como `capacidad` y un método `moverse`. Luego crea clases hijas `Autobus` y `Avion` que sobrescriban el método `moverse` para representar diferentes formas de transporte. Practica herencia y polimorfismo.

1.6 Ejercicio 6: Gestión de Usuarios

Crea una clase `Usuario` con atributos privados como `nombre` y `contrasena`, y métodos públicos para modificar esos atributos. Implementa una clase hija `Admin` que añada un método `eliminarUsuario`. Practica encapsulamiento y herencia.

1.7 Ejercicio 7: Empleados y Salarios

Crea una clase base `Empleado` con un método `calcularSalario()`. Implementa clases hijas `EmpleadoPorHora` y `EmpleadoAsalariado`, cada una con su propia forma de calcular el salario. Practica herencia y polimorfismo al llamar al método `calcularSalario()` en una lista de diferentes empleados.

1.8 Ejercicio 8: Zoológico

Crea una clase abstracta `Animal` con un método abstracto `hacerSonido()`. Luego implementa clases hijas `Perro`, `Gato` y `Pajaro` que sobrescriban `hacerSonido()` de forma diferente. Practica abstracción y polimorfismo.

1.9 Ejercicio 9: Sistema de Gestión de Productos

Diseña una clase `Producto` que tenga atributos como `nombre` y `precio`. Implementa clases hijas como `ProductoElectronico` y `ProductoAlimenticio`, cada una con su propio método `calcularImpuesto()`. Practica herencia y encapsulamiento.

1.10 Ejercicio 10: Juegos de Mesa

Crea una clase `JuegoDeMesa` con métodos `iniciar()` y `finalizar()`. Luego, implementa clases hijas como `Ajedrez` y `Monopolio`, cada una sobrescribiendo el comportamiento de `iniciar()` y `finalizar()`. Practica herencia y polimorfismo.