Trabajo Práctico 3 - Introducción a la Programación Orientada a Objetos (POO)

EMILIANO JARA

43.592.980

Objetivo General

Aplicar los fundamentos de la Programación Orientada a Objetos (POO) en Java, incluyendo el diseño de clases, creación de objetos, uso de atributos y métodos, encapsulamiento, manejo de estado e identidad, y reutilización de código.

Marco Teórico

La Programación Orientada a Objetos (POO) constituye uno de los pilares fundamentales en el desarrollo moderno de software. Este paradigma permite representar entidades del mundo real como objetos que poseen atributos (estado) y métodos (comportamiento), facilitando la creación de sistemas modulares, escalables y reutilizables.

El presente trabajo práctico tiene como objetivo aplicar los conceptos esenciales de la POO utilizando el lenguaje Java, abordando el diseño de clases, la creación de objetos, el uso de modificadores de acceso, la implementación de métodos, y la gestión del estado e identidad de los objetos. A través de cinco actividades concretas, se busca modelar situaciones reales mediante clases que simulan comportamientos específicos, como el registro de estudiantes, el envejecimiento de mascotas, la validación de datos en libros, la gestión de gallinas en una granja digital, y la simulación de una nave espacial.

Cada actividad fue desarrollada respetando los principios de encapsulamiento, abstracción y modularidad, promoviendo buenas prácticas de programación y reforzando el pensamiento lógico y orientado a objetos. El trabajo se complementa con pruebas en el método main() para verificar el funcionamiento de cada clase, asegurando que los objetos respondan de manera coherente a las acciones que se les aplican.

Actividades del Trabajo Práctico

Actividad 1: Registro de Estudiantes

Modelar una entidad Estudiante aplicando los principios básicos de la Programación Orientada a Objetos (POO), como encapsulamiento, estado, identidad y comportamiento. Esta actividad busca representar un estudiante con sus datos personales y académicos, y permitir la modificación controlada de su calificación.

Actividad 2: Registro de Mascotas

Modelar una entidad Mascota que represente un animal doméstico, aplicando los principios de la Programación Orientada a Objetos (POO). Esta actividad busca simular el paso del tiempo en la vida de una mascota, permitiendo observar cómo cambia su estado interno (edad) y cómo se mantiene su identidad.

Actividad 3: Encapsulamiento con la Clase Libro

Aplicar el principio de **encapsulamiento** en el diseño de una clase Libro, restringiendo el acceso directo a los atributos y gestionando su modificación mediante métodos controlados. Esta actividad busca reforzar el uso de **getters** y **setters**, y la validación de datos para proteger la integridad del objeto.Se probó el comportamiento en el método main con intentos válidos e inválidos de modificación.

Actividad 4: Gestión de Gallinas en Granja Digital

Modelar una entidad Gallina que represente un animal de granja, aplicando los principios de la Programación Orientada a Objetos (POO). Esta actividad busca simular el comportamiento de las gallinas en un entorno digital, permitiendo que envejezcan, pongan huevos y se pueda consultar su estado actual.

Actividad 5: Simulación de Nave Espacial

Se creó la clase NaveEspacial con atributos privados: nombre y combustible. Se implementaron métodos para despegar, avanzar, recargar combustible y mostrar estado. Se incluyeron validaciones para evitar avanzar sin suficiente combustible y para no superar el límite al recargar.

Conclusión

El desarrollo del Trabajo Práctico 3 permitió aplicar de manera concreta los fundamentos de la **Programación Orientada a Objetos (POO)** en el lenguaje Java, abordando conceptos esenciales como **clases, objetos, atributos, métodos, encapsulamiento, estado, identidad y reutilización de código**.

A lo largo de las cinco actividades propuestas, se modelaron entidades del mundo real —como estudiantes, mascotas, libros, gallinas y naves espaciales— mediante clases bien estructuradas,

cada una con sus propios atributos y comportamientos. Esta representación permitió simular situaciones prácticas y dinámicas, reforzando el enfoque modular y reutilizable que caracteriza a la POO.

El uso de **encapsulamiento** fue clave para proteger los datos internos de cada objeto, evitando accesos directos y promoviendo el uso de **getters y setters** para una gestión segura de los atributos. Además, se incorporaron **validaciones lógicas** en los métodos, especialmente en la clase Libro y NaveEspacial, lo que permitió controlar el estado del objeto y evitar inconsistencias.

Cada clase fue probada mediante el método main(), lo que permitió verificar su funcionamiento, simular acciones y observar cómo los objetos evolucionan manteniendo su identidad. Esta práctica consolidó el entendimiento de cómo los objetos interactúan en un sistema y cómo su estado puede cambiar de forma controlada.