Assignment 2 Journal: Object-Oriented Problems

Boris Bojanov

ID: 3608550

Date Started: Oct 5 2024
Date Completed: Oct 29 2024

Initial Thoughts and Planning

Assignment 2 focused on applying object-oriented programming (OOP) concepts in C++ through five problems. These included class hierarchies, inheritance, and encapsulation. I planned each problem by determining the most suitable OOP features to use.

Problem 1: Animal Class and Inheritance

Process and Challenges

I created a base `Animal` class with a `sound()` method and derived `Pig`, `Sheep`, `Duck`, and `Cow`, each overriding `sound()`. The `AnimalTest` class instantiated animals based on user input, adding an interactive element.

Reflections

This problem illustrated inheritance and code reusability by extending a generic base class with specific subclasses.

Problem 2: Book Class with Access Control

Process and Challenges

The `Book` class used private attributes with public `get` and `set` methods. Testing multiple `Book` objects verified proper encapsulation.

Reflections

Encapsulation ensured controlled data access, highlighting the importance of restricting direct attribute access.

Problem 3: Elevator Class and Cleanup

Process and Challenges

The `Elevator` class simulated an elevator, with constructors to set building height and a `finalize()` method to return the elevator to the first floor. Testing scenarios ensured expected behavior.

Reflections

This problem emphasized cleanup and resource management through object destruction.

Problem 4: Rodent Inheritance Hierarchy

Process and Challenges

I created a `Rodent` base class and derived `Mouse`, `Gerbil`, `Hamster`, and `GuineaPig`, refining behaviors like `eat` and `sleep` in each subclass.

Reflections

This problem reinforced inheritance and refining base class behaviors, demonstrating how polymorphism can adapt code.

Problem 5: Point and Shape Hierarchy

Process and Challenges

I created a `Point` class, followed by a `Shape` base class with methods for area, circumference, and bounding boxes. Derived classes (`Circle`, `Rectangle`, `Triangle`) implemented specific constructors and calculations, with error-checking for robustness.

Reflections

This problem demonstrated polymorphism and the power of managing different shapes through a common interface.

Summary Reflections

Assignment 2 provided practice with OOP concepts like inheritance, encapsulation, and polymorphism. Each problem highlighted different OOP aspects, improving code structure and reusability. Keeping this journal helped me reflect on my progress and the effectiveness of OOP design.

Sources

https://devdocs.io/cpp/