**Parking System Classes**

ICT 4305: Object Oriented Methods and Programming

Michelle Agustin

University of Denver College of Professional Studies

October 5, 2025

Faculty: Nirav Shah, M.S , MBA

Director: Cathie Wilson, M.S.

Dean: Michael J. McGuire, MLS

Working on this assignment was both challenging and rewarding. I found the most challenging part to be designing the relationships between the Customer, Car, and ParkingLot classes in a way that was both logical and flexible for future updates. Deciding how much logic should belong in each class took some thought, for example, whether a Customer should directly register a Car or if that should be handled by another class such as a permit management system. I ultimately chose to keep register() inside the Customer class since it made sense that the customer initiates vehicle registration.

The easier parts were implementing the individual data classes like Address and the enumeration CarType. These were straightforward because they mainly required storing and returning data, and Java makes working with enum types intuitive. Using the java.time.LocalDate class for permit expiration was also much easier than older date-handling options like java.util.Date or Calendar. The modern time API is more readable, type-safe, and eliminates a lot of common date-related bugs.

What helped me most was breaking the problem down step by step and testing each class separately as I went. After implementing one class, I wrote quick test methods to verify that constructors, getters, and other methods behaved correctly. This approach made debugging easier, and I caught small issues like forgetting to initialize a field, before they caused larger errors later on. The use of test-driven thinking helped me refine my design before I finalized it.

I wish I had known more about creating UML diagrams before starting. Understanding how to represent associations and multiplicities properly would have made it easier to plan before writing the code. The diagrams also clarified how each class interacted and helped ensure I wasn’t duplicating responsibilities.

In terms of implementation decisions, I added constructors and getter methods even though the UML sketch didn’t require them, since they make the code testable and functional. I also chose to include validation logic in certain places like preventing duplicate car entries in a full parking lot, to make the application more realistic. Overall, this assignment helped me strengthen my understanding of Java class design, relationships, and practical testing.

**References**

Bloch, Joshua. 2018. Effective Java. 3rd ed. Boston: Addison-Wesley.

Oracle. 2024. “Trail: Learning the Java Language – Classes and Objects.” The Java Tutorials. Oracle. <https://docs.oracle.com/javase/tutorial/java/javaOO/>.