Object-Oriented Programming
Semester 2025-III
Workshop No. 2 — Object-Oriented Implementation

Eng. Carlos Andrés Sierra, M.Sc.
Full-time Adjunct Professor
Computer Engineering Program
School of Engineering
Universidad Distrital Francisco José de Caldas

Building on *Workshop #1* (Object-Oriented Design), this session focuses on translating your **conceptual design** into a technical implementation plan. You will refine your understanding of class structures, relationships, and the key OOP pillars (**inheritance**, **polymorphism**, **encapsulation**) to move closer to a working prototype of your *simple transactional application*.

Workshop Scope and Objectives:

- **Refinement of Conceptual Design:** Update your original requirements and user stories based on feedback or new insights obtained since Workshop #1.
- **Technical Design with UML:** Produce and refine *UML class diagrams* that detail inheritance hierarchies, key associations, and class responsibilities.
- **OOP Principles in Action:** Demonstrate how inheritance, polymorphism, and encapsulation are applied to solve realistic scenarios in your transactional application.

Carlos Andrés Sierra, Computer Engineer, M.Sc. in Computer Engineering, Titular Professor at Universidad Distrital Francisco José de Caldas.

Any comment or concern regarding this workshop can be sent to Carlos A. Sierra at: *cavirguezs@udistrital.edu.co*.

Methodology and Deliverables:

1. **Conceptual Design Updates:**
   - Revisit *Workshop #1* deliverables (requirements, CRC cards, user stories) and add any missing or revised elements.
   - Document changes or decisions that emerged from further planning or discussions.
2. **Technical Design (UML Diagrams):**
   - Create or update *class diagrams*, showing inheritance hierarchies, class members (attributes, methods), and relationships.
   - Highlight how classes implement or override methods to fulfill user requirements and integrate the OOP principles.
   - Add *sequence diagrams* or *activity diagrams* to illustrate interactions between classes and the flow of data.
3. **Implementation Plan for OOP Concepts:**
   - Summarize how your code will realize **encapsulation** (access modifiers, getters, setters), **inheritance** (base and derived classes), and **polymorphism** (overriding, overloading, or interfaces).
   - Outline a preliminary directory structure or package organization that supports your design.
4. **Work in Progress Code:**
   - Include initial code snippets illustrating class definitions, placeholder methods, or abstract classes.
   - Clearly state how these snippets relate to your UML diagrams and design rationale.
5. **Delivery Format:**
   - Archive your updated documentation (revised conceptual design, UML diagrams, any code snippets) into a single PDF.
   - Place all materials in a folder named `Workshop-2` in your project repository, referencing them in a `README.md`.

Deadline: Friday, October 17th, 2025, 16:00. Submissions after this time may be penalized.

Notes:

- Keep your documents in **English**.
- Cite any relevant tutorials, articles, or external libraries that shaped your design.

- Your refined design and UML models here lay the groundwork for actual coding in future assignments. Aim for clarity and scalability.
- Deliveries are incremental: build on your previous workshop and ensure each submission shows progress and refinement.
- Consider including a brief reflection (1-2 paragraphs) describing challenges faced and decisions made during this phase.

In this workshop, you turn theoretical plans into a technical blueprint. Focus on integrating OOP concepts rigorously, ensuring your transactional application remains robust, maintainable, and aligned with best practices.