UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 1

Object-oriented Programming

Submitted by:
Catahan, Joshua A.

Instructor:
Engr. Maria Rizette H. Sayo

07, 26, 2025

I. Objectives

This laboratory activity aims to implement the principles and techniques in object-oriented programming specifically through:

- Identifying object-orientation design goals
- Identifying the relevance of design pattern to software development

II. Methods

- Software Development
    - The design steps in object-oriented programming
    - Coding style and implementation using Python
    - Testing and Debugging
    - Reinforcement of below exercises

A. Suppose you are on the design team for a new e-book reader. What are the primary classes and methods that the Python software for your reader will need? You should include an inheritance diagram for this code, but you do not need to write any actual code. Your software architecture should at least include ways for customers to buy new books, view their list of purchased books, and read their purchased books.

B. Write a Python class, Polygons that has three instance variables of type str, int, and float, that respectively represent the name of the polygon, its number of sides, and its area. Your class must include a constructor method that initializes each variable to an appropriate value, and your class should include methods for setting the value of each type and retrieving the value of each type.

III. Results

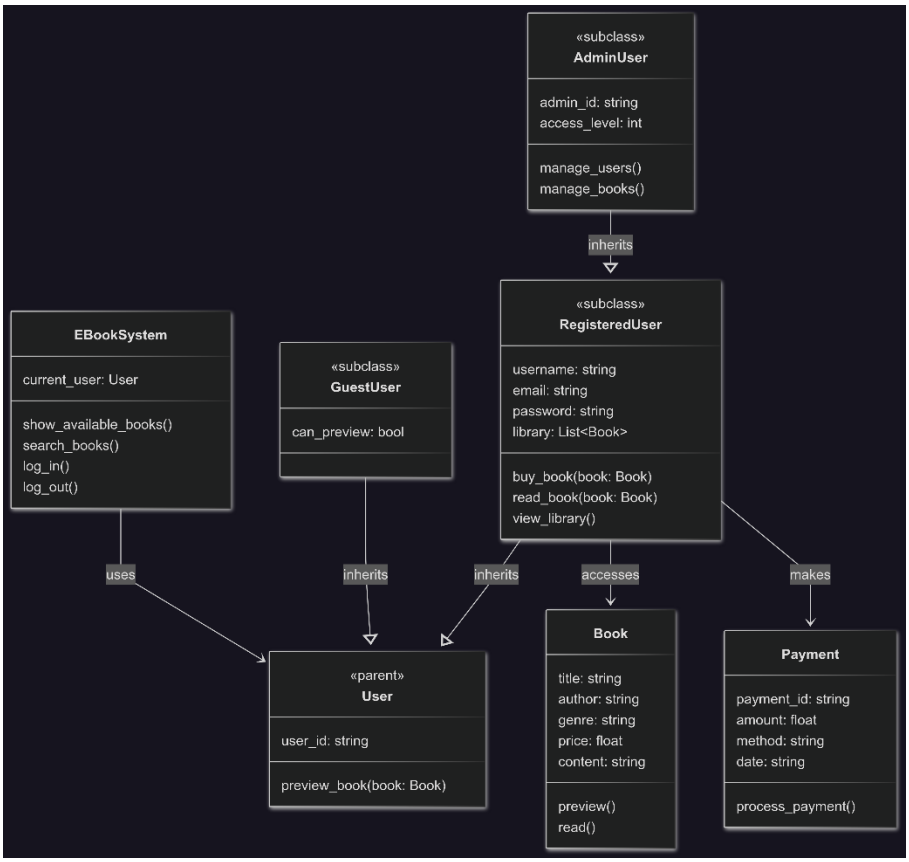

Figure 1: Diagram

For this task, I designed a software architecture for an e-book reader using object-oriented programming (OOP). I applied inheritance to create a user hierarchy and defined how users interact with books and payments in the system.

The core class is EBookSystem, which manages user sessions and lets users search or browse books. A base class called User defines common actions like previewing books. From it, the RegisteredUser class inherits and expands functionality by adding the ability to buy and read books and maintain a personal library. Finally, AdminUser inherits from RegisteredUser, meaning that admins have all user features plus management tools like editing users or books. This hierarchical inheritance allows us to reuse common features while extending functionality for more advanced user roles. This concept is common in real-world platforms such as Kotobee, where different user levels (guest, reader, admin) have layered permissions [2][3].
This inheritance enables us to build more scalable and maintainable systems by reducing code duplication and making relationships between objects clearer [2].

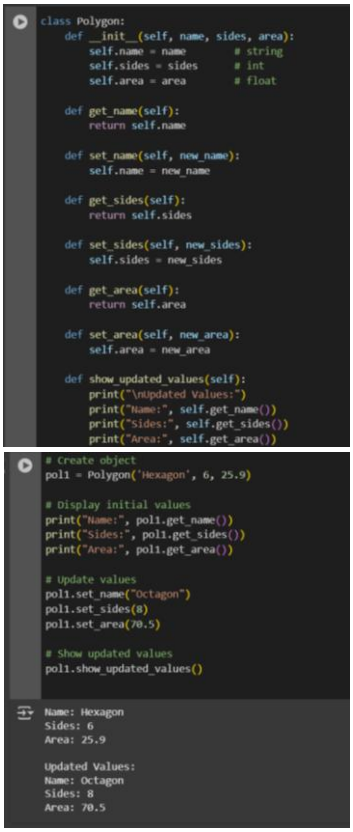

Figure 2: Screenshot of program

In this part of the lab, I created a Python class called Polygon. It uses three instance variables: a string for the name of the polygon, an integer for the number of sides, and a float for the area. These values are initialized using a constructor so that I can set them when I create an object. To protect the data inside the class, I used getter and setter methods instead of allowing direct access. This follows the concept of encapsulation, which keeps data secure and prevents unwanted changes from outside the class. In which according to GeeksforGeeks, encapsulation improves the flexibility and security of a program because it controls how data is accessed [1]. Additionally, by using functions like setters, I was able to modify the values assigned to an object easily. By creating the method show_updated_values(), I was able to avoid writing multiple lines of repetitive code when dealing with many objects. For example, without this method, I would have to write print() repeatedly, as shown in the "Display initial values" section in the figure above. This demonstrates one of the many advantages of OOP.

IV. Conclusion

In this laboratory report, I was able to refresh and strengthen my understanding of Python Object-Oriented Programming. From creating a diagram that shows the connection between classes through inheritance to implementing a program that demonstrates how classes, methods, and functions work in Python. To summarize, inheritance helped us manage different types of users in the e-book system, while encapsulation kept our Polygon class secure and organized. These are core OOP principles that make software easier to understand, reuse, and maintain.

References

[1] GeeksforGeeks, “Encapsulation in Java,” *GeeksforGeeks*, [Online]. Available: https://www.geeksforgeeks.org/java/encapsulation-in-java. [Accessed: Jul. 28, 2025].

[2] Algocademy, “Understanding Inheritance in Object-Oriented Programming: A Comprehensive Guide,” *Algocademy*, Jul. 2023. [Online]. Available: https://algocademy.com/blog/understanding-inheritance-in-object-oriented-programming-a-comprehensive-guide/. [Accessed: Jul. 28, 2025].

[3] A. Hussein, “How to Build Your Own eBook Platform,” *Kotobee Blog*, Aug. 2021. [Online]. Available: https://blog.kotobee.com/ebook-platforms/. [Accessed: Jul. 28, 2025].