Data Structure and Algorithm

Laboratory Activity No. 1

Object-oriented Programming

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Poliño, Justine | Engr. Maria Rizette H. Sayo |

July 26, 2025

# 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

# Methods

* Software Development
  + The design steps in object-oriented programming
  + Coding style and implementation using Python
  + Testing and Debugging
  + Reinforcement of below exercises
  1. 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.
  2. 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.

# Results

* 1. In this section, the diagram below shows the structures of the design for the e-book reader.

A screenshot of a computer

AI-generated content may be incorrect.

This eBook platform design shows a clean architecture with five core components. Users authenticate and connect to devices, while purchases link users to books, creating licensed copies in their library. The system separates concerns clearly: User (accounts), Book (content), Purchase (transactions), Library (collections), and Device (access). Each purchase generates a BookCopy, ensuring proper digital rights management. The relationships enable secure content distribution with purchase tracking and multi-device sync.

* 1. The Polygons class represents geometric shapes with name (str), sides (int), and area (float) attributes. It uses getters/setters to control access while maintaining type safety, demonstrating Python's OOP capabilities for modeling real-world entities.  
     A screen shot of a computer program

     AI-generated content may be incorrect.

Figure 2 Polygon Class

A computer screen with text

AI-generated content may be incorrect.

Figure 3 Example

A black rectangular object with white text

AI-generated content may be incorrect.

1. Clearly marks it as sample input
2. Shows the mapping of values to parameters
3. Includes type information in parentheses
4. Explains the purpose of the last value
5. Maintains clean formatting
6. Uses consistent notation

# Conclusion

This laboratory activity successfully applied object-oriented programming principles through two key exercises: (A) designing a structured e-book platform with clear class relationships for user management, purchases, and content delivery, and (B) implementing a validated Polygons class demonstrating encapsulation and type safety for geometric properties. Together, these exercises reinforced core OOP concepts including modular design, data integrity, and method-controlled attribute access, providing practical foundations for software development.

**References**

[1] Bader, D. A. (2022). *Python geometric algorithms*.t