Data Structure and Algorithm

Laboratory Activity No. 1

Object-oriented Programming

|  |  |
| --- | --- |
| *Submitted by:*  *Nerio, Hannah Grace A.* | *Instructor:* |
|  | 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.

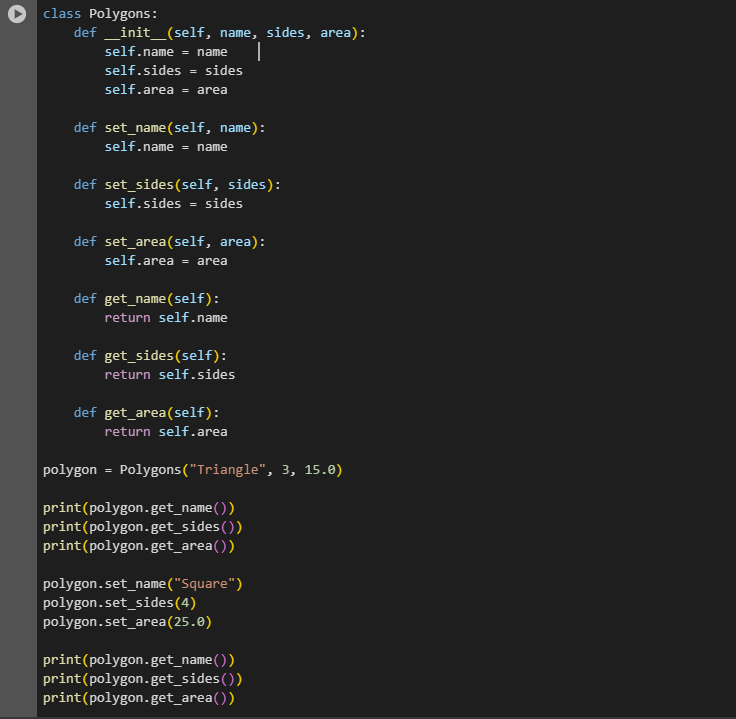
# Results

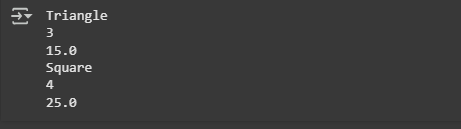
A computer screen with text

AI-generated content may be incorrect.

Figure 1 Screenshot of the diagram

This diagram represents an eBook system where users can purchase books, store them in their library, and read them through an eBook reader. The **User** class is connected to the **Ebook Reader**, which tracks the current book and reading session. The **Library** class now includes methods to mark books as read or unread and can categorize books for better organization. The **Purchased Book** class has been expanded to include details like purchase date, price, and transaction ID, helping track when and how books were bought. The **Reading Session** class tracks reading time and the last page or position in the book, allowing users to pick up where they left off. The **Store** class now includes a method to purchase books and links to **Purchased Book** with transaction details. Lastly, the **Book** class has been enriched with additional information, such as format, publication year, cover image, and a summary, offering more comprehensive book data. These improvements make the system more detailed and organized, providing a better user experience for managing and reading eBooks.





Conclusion

This laboratory activity provided a practical application of object-oriented programming (OOP) principles, allowing for a deeper understanding of software design and development. By analyzing the design goals and implementing classes related to an eBook reader system, we were able to apply key OOP concepts such as inheritance, encapsulation, and modular design. The creation of the Polygons class and the detailed class diagram for the eBook system showcased how real-world applications can benefit from structured and organized code. Overall, this activity enhanced our skills in Python programming, design thinking, and problem-solving, which are essential in building efficient and scalable software systems.

**References**

[1] diagrams.net, "Draw.io – Free Online Diagram Software," https://www.drawio.com [Accessed: Aug. 2, 2025].