Data Structure and Algorithm

Laboratory Activity No. 1

Object-oriented Programming

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Acebedo, Sebastian C. | 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

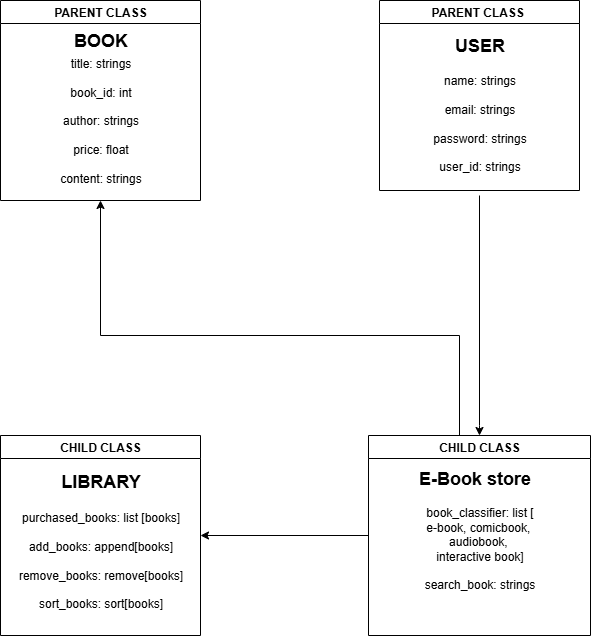


Figure 1 Screenshot of program

This is a class diagram representing the structure of a book system, featuring parent classes "Book" and "User," and their child classes "Library" and "E-Book Store." It showcases attributes and methods for managing books, users, and purchases within the system.

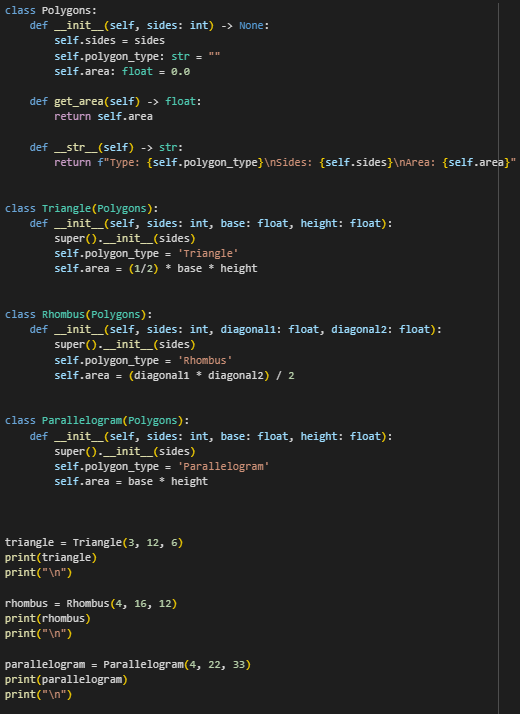


Figure 2 Screenshot of program

This is a Python program demonstrating inheritance and polymorphism using a class hierarchy for different polygon shapes. The Polygons class is the parent class, with child classes Triangle, Rhombus, and Parallelogram each inheriting from it. Each subclass calculates and prints the area specific to its shape. The output shows the type, number of sides, and area for each polygon.

Reference link:

[DSA-Lab-Report-1.ipynb - Colab](https://colab.research.google.com/drive/1eAZhjNMLA0uPOBzDcpNsjtKr8Ai-qYLS?authuser=0#scrollTo=iuRTzuWIBRYo)

Conclusion

In Python, classes help organize code by bundling data and methods together, making it more readable, reusable, and easier to maintain. They allow us to create objects that represent real-world concepts, simplifying complex problems.

Learning class diagrams is important because they visually show how classes relate to each other. They help us plan our code, spot issues early, and communicate ideas clearly. Understanding both classes and class diagrams is essential for writing clean, efficient code and for real-world software development.

**References**

[1] Co Arthur O.. “University of Caloocan City Computer Engineering Department Honor Code,” UCC-CpE Departmental Policies, 2020.

[2] L. Tagliaferri, “Understanding class inheritance in Python 3,” *DigitalOcean*, Aug. 20, 2021. <https://www.digitalocean.com/community/tutorials/understanding-class-inheritance-in-python-3>

[3] A. User, “Python Parent class and inheritance,” *askthedev.com*, Sep. 29, 2024. https://askthedev.com/python-parent-class-and-inheritance/