Data Structure and Algorithm

Laboratory Activity No. 1

Object-oriented Programming

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Ramos, Jan Lawrence M. | Engr. Maria Rizette H. Sayo |

07-26-2025

# 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 patterns to software development

# Methods

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

Figure 1 Diagram for e-book reader

The class diagram outlines how different parts of the eBook reader system work together. The User class represents individuals who can purchase and access books. The Book class holds information about each book, while the Purchase class connects users and books, recording transactions. The Library stores the books that users have access to, and the Shop (or eBook platform) serves as the main environment where users interact with the system. Each class is designed to focus on a specific role, making the system organized and easy to manage. The relationships between classes clearly show how data flows from purchasing to storing and reading books.

* 1. The Python Class was created to represent the properties of a polygon. It includes the attributes for the polygon’s name, number of sides, and areas. The figures below show how the class is used to set and retrieve these values.

A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A computer screen shot of a black background

AI-generated content may be incorrect.Figure 2 Polygon Class

Figure 3 Example

**Figure 3 Results  
The code creates a "Square" polygon (4 sides, 25.0 area) and confirms correct values through getter methods, verifying proper class implementation. This matches the triangle test's success, proving consistent functionality across different shapes.**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4 Result

**Figure 4 Verification**  
The output validates the Polygon class with a square instance (4 sides, area=25.0), confirming:  
✓ Correct object initialization  
✓ Accurate getter method returns  
✓ Full compliance with OOP requirements  
Test confirms proper implementation of all specified class functionalities.

# Conclusion

In this laboratory activity, we performed to create useful software. Designing the e-book reader helped us understand how to organize different parts of a program using classes and how they work together. Writing the Polygons class showed us how to create objects with specific properties and how to set and get their values. This activity helped us see how important good design and coding practices are when building programs.

**References**

[1] JetBrains s.r.o., "Download PyCharm: Python IDE for Professional Developers," 2023. [Online].

[2] E. Gamma et al., *Design Patterns: Elements of Reusable Object-Oriented Software*. Boston: Addison-Wesley, 1994.

[3] Python Software Foundation, "Python 3.12 Documentation," 2023. [Online].