

**FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY**

page1image9370432

**PROG211 – Objected Oriented Programming Method1**

page1image9370432

Title : Individual Assignment

Issue Date : Week 2

Due Date : Week 4

Lecturer/Examiner :Mr. Elijah Fullah

Name of Student/s :Alhaji AbuBakarr Jalloh

Student ID No. : 905005263

Class : DIT 1101F

Year/Semester : 2/1

Academic Honesty Policy Statement

page1image9370432

I/We, hereby attest those contents of this attachment are my own work. Referenced works, articles, art, programs, papers or parts thereof are acknowledged at the end of this paper. This includes data excerpted from CD-ROMs, the Internet, other private networks, and other people’s disk of the computer system.

Student’s Signature: Date: 18TH OCTOBER 2025

page1image9370432

LECTURER’S COMMMENTS/GRADE:

for office use only upon receive

Remark

DATE:  
 TIME:

RECEIVER’S NAME:

**Table of Contents**

1. Introduction.....................................................

2. Data Structure Selection ...................................

3. Core Functionality & Validation ........................

4. UML Diagram............................

5. Testing Strategy ..................................................

6. Conclusion........................................................

7. References .........................................................

**1. Introduction**

This document explains the design decisions behind the Mini Library Management System developed for Assignment 1.

The system is implemented in pure Python using only built-in data structures (dictionaries, lists, and tuples) to manage books and members. It supports full CRUD operations (Create, Read, Update, Delete) along with borrowing and returning books, with strict validation rules. The design focuses on simplicity , efficiency, correctness, and full compliance with the assignment brief.

**2. Data Structure Selection**

Books are stored in a dictionary where the ISBN is the key and the value is a tuple containing title, author, genre, and total copies. This choice enables fast O(1) lookup for adding, updating, deleting, and checking availability, essential for a library system. The tuple ensures that book details remain immutable once added, preventing accidental changes.

Members are stored in a list of tuples, where each tuple contains the member ID, name, email, and a mutable list of borrowed ISBNs . The list allows easy iteration and appending of new members, while the nested list supports dynamic borrowing and returning.

Valid genres are stored in an immutable tuple `("Fiction", "Non-Fiction", "Sci-Fi")`. This prevents runtime modification and guarantees that only approved genres are accepted.

**3. Core Functionality & Validation**

All functions include clear error messages and return `True` or `False` for testability.

* `add\_book()` checks for unique ISBN, valid genre, and non-negative copies before insertion.
* `add\_member()` ensures the member ID is unique.
* `borrow\_book()` enforces:
* Maximum 3 books per member
* No duplicate borrowing of the same book
* At least one copy available
* `return\_book()` verifies the member actually borrowed the book before updating counts.
* `delete\_book()` is only allowed if no copies are currently borrowed.
* `delete\_member()` is only allowed if the member has no borrowed books\*\*.
* `update\_book()` supports partial updates (e.g., change only title) and validates genre and copy count.

A helper function `available(isbn)` calculates real-time free copies by subtracting borrowed count from total copies.

**5. Testing Strategy**

The system includes 6+ unit tests in `tests.py` using `assert` statements. Each test resets the global state before running to ensure isolation. Tests cover:

- Adding a valid book and detecting duplicate ISBNs

- Borrowing when copies are available vs. when none are left

- Enforcing the maximum 3 books per member rule

- Preventing deletion of a borrowed book

- Preventing deletion of a member with borrowed books

- Verifying that returning a book increases availability

Running `python tests.py` outputs: ALL TESTS PASSED

**6. Conclusion**

The Mini Library Management System is a robust, efficient, and fully validated solution that meets all assignment requirements. The use of dictionaries for fast access, tuples for immutability, and lists for flexibility ensures correct behavior and easy maintenance. The system is thoroughly tested and well-documented with a clear UML diagram. This approach ensures the system is both functional and maintainable within the given constraints.

7. References

- Python Software Foundation. (2025). Data Structures. Python Documentation. Available at: https://docs.python.org/3/tutorial/datastructures.html

- Python Software Foundation. (2025). Tuples and Sequences. Available at: https://docs.python.org/3/library/stdtypes.html#tuples

- Real Python. (2025). Python Lists and Tuples. Available at: https://realpython.com/python-lists-tuples/