

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

page1image9370432

PROG211 – OBJECTED ORIENTED PROGRAMMING METHODS 1

page1image9370432

Title : Individual Assignment

Issue Date : Week 2

Due Date : Week 4

Lecturer/Examiner : Mr.Elijah Fullah

Name of Student/s : David B Collier

Student ID No. : 905005212

Class : DIT 1101F

Year/Semester : 2/3

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: 26 OCTOBER 2025

page1image9370432

LECTURER’S COMMMENTS/GRADE:

for office use only upon receive

Remark

DATE:  
 TIME:

RECEIVER’S NAME:

**Design Rationale**

**Introduction**

The **Lumley Community Library Management System** was developed to efficiently manage the day-to-day operations of a community library, including the registration of users, the cataloging of books, and the borrowing and returning of materials. In designing the system, it was important to ensure that the data structures chosen could handle these operations in a way that is both efficient and easy to maintain. The system needs to support quick searches for books, keep track of user activities, and categorize books accurately according to their genres.

To achieve this, different types of data structures were selected based on their strengths and suitability for the specific tasks: dictionaries for book management, lists for user management, and tuples for storing fixed categories of genres. These choices were made to optimize performance, clarity, and data integrity, ensuring that the system remains responsive and reliable even as the number of users and books grows.

**Why a Dictionary for Books (lumley\_books)**

Books in a library have many details—like the title, author, type, total copies, and how many are available. I used a dictionary to store books because each book has a unique code (book\_code) which works perfectly as a key.

* **Quick Access:** When someone wants to find a book, the system can look it up immediately using its code. This is much faster than searching through a list.
* **Organized Data:** Each book’s details are stored neatly under one key, making it easy to manage and update.
* **Easy Updates:** Functions like alter\_book() or lend\_book() can quickly change a book’s information without scanning the whole collection.

Using a dictionary makes handling books efficient and straightforward, even if the library grows in size.

**Why a List for Users (lumley\_users)**

Users are registered over time, and their order of registration can be important. That’s why a list was chosen for storing users.

* **Flexible Growth**: New users can be added easily with append() without worrying about resizing the structure.
* **Maintains Order**: Users remain in the order they were added, which can be useful for reports or administrative tasks.
* **Easy to Iterate**: When checking who has borrowed books or displaying user information, a list makes it simple to go through all entries.

Lists are perfect here because they handle dynamic and sequential data naturally.

**Why a Tuple for Genres (lumley\_genres)**

The library has a fixed set of genres: Fiction, Non-Fiction, and Sci-Fi. Since these categories don’t change, a **tuple** was used.

* **Unchangeable:** Tuples cannot be modified, which prevents accidental changes to genre names.
* **Memory Efficient:** Tuples are more memory-efficient than lists, which is beneficial when handling fixed sets of data that do not require modification.
* **Clear Purpose:** By using a tuple, it is clear to anyone reading the code that genres are constant values, reinforcing the idea that they represent a fixed category set within the library system.

**Conclusion**

The combination of **dictionary, list, and tuple** was chosen deliberately to leverage the strengths of each data structure:

* **Dictionaries** provide fast access and flexible key-value storage for books.
* **Lists** offer dynamic, ordered collections suitable for user management.
* **Tuples** provide immutable, fixed collections for genres.

These choices make the Lumley Community Library Management System not only efficient but also easy to read, maintain, and scale. By matching the right data structure to each part of the system, everyday operations like adding books, registering users, and lending items are handled smoothly and reliably.

**UML DIAGRAM SKETCH**

