# DPIT121 Assignment 1: Object-Oriented Design and Programming in Python

***Due Date: Week 5   
Total Marks: 10 (Task 1 – 5 marks and Task 2 - 5 marks)***

# Objective

The objective of Assignment 1 is to apply the knowledge of object-oriented design and programming learned from Week 1 to Week 4 lectures. This project aims to create a simple library management system with classes for handling books, members, and library operations. You will design appropriate classes using object-oriented concepts, UML class diagrams, and the corresponding Python code.

# Assignment Description

This Python code should represent a simple library management system composed of four main classes: LibraryItem, Book, Member, and Library, along with a main function to interact with the system through a command-line interface.

* **LibraryItem class**: Acts as a base class for items that can be found in a library. It stores a title attribute common to all library items.
* **Book class**: mInherits from LibraryItem and adds specific attributes for books, including id, author, and availability to track if the book is available for borrowing.
* **Member class**: Represents a library member with a member\_id, name, and a list of borrowed\_books to keep track of which books the member has borrowed.
* **Library class**: Manages the collection of books and members. It has methods to:
  + Display the entire book collection and check their availability.
  + Add new books to the collection.
  + Search for books by title or ID (either way is acceptable).
  + Display the list of current members.
  + Add new members.
  + Allow members to borrow and return books, updating the availability status of the books and the list of books a member has borrowed.

The main function provides a text-based menu system to interact with the library management system, allowing users to perform actions such as viewing the book collection, adding books, searching for books by name, managing members, borrowing and returning books, and exiting the program. The system runs in an infinite loop, prompting the user for their choice of action until they decide to exit.

# Detailed Class and Method Description

* Design suitable classes using class inheritance to represent the hierarchical structure of the library system. (You will be creating a UML diagram representing the design of the program)
* Design class attributes to store information about books, member information and borrowing.
* Implement effective class methods using polymorphism to record and search for information based on user inputs and requirements as shown in the UI
* **You must include 5 book objects and 5 member objects within the code as test data within the main method.**

**Library Item Class**

* **Attributes**:
  + title: Stores the title of the library item.
* **Methods**:
  + \_\_init\_\_(self, title): Initializes a new library item with its title. This class serves as a foundation for more specific types of library items, allowing for future expansion beyond books if needed.

**Book Class**

* **Inheritance**: Inherits from LibraryItem.
* **Attributes**:
  + id: A unique identifier for the book.
  + author: The author of the book.
  + available: A boolean indicating whether the book is currently available for borrowing.
* **Methods**:
  + \_\_init\_\_(self, id, title, author): Besides inheriting the title from LibraryItem, it initializes the book with a unique id, author, and sets available to True.

**Member Class**

* **Attributes**:
  + member\_id: A unique identifier for the library member.
  + name: The name of the library member.
  + borrowed\_books: A list to keep track of the books borrowed by the member.
* **Methods**:
  + \_\_init\_\_(self, member\_id, name): Initializes the member with a member\_id, name, and an empty list for borrowed\_books.

**Library Class**

* **Attributes**:
  + books: A list to store all books in the library.
  + members: A list to store all members of the library.
  + next\_book\_id: An integer to keep track of the next unique book ID for new books.
* **Methods**:
  + Various methods for displaying books and members, adding new books and members, searching for books by name or ID, searching for members by ID, borrowing, and returning books, as previously described.

**User Interface (UI) Description**

The UI is command-line-based, guiding the user through various library management tasks. Here are the UI (Main Menu) options:

1. **Main Menu**: Presents nine options to the user when the program is run and processes input to navigate to the desired function.

1. Show book collection

2. Add book

3. Search book by book name

4. Show list of current members

5. Add new member

6. Search member by member id

7. Borrow book

8. Return book

9. Exit

1. **Input Prompts**: Asks the user for necessary information, like book titles, author names, member IDs, depending on the chosen operation within each option available within the UI.
2. **Validation and Error Handling**: Optional and input validation is not necessary in this assignment. No marks will be deducted for lack of input validation in this assignment.
3. **Feedback Messages**: Confirms successful operations (e.g., book addition, book borrowing/returning) or informs the user of issues (e.g., book not available, member not found).

# Task 1 Submission – Report (Total 5 marks)

Create a PDF report named **"Assignment1\_class\_design.pdf"** that includes the following:

* 1. **~~Description of the object-oriented design~~** ~~including attributes and methods of each class and the relationships between different classes.~~ ***~~(1 mark)~~***
  2. **UML class diagram** with relationship drawn using professional tools like UMLet (hand-drawn diagrams are not acceptable). ***(1 marks)***
  3. ~~Minimum following Screenshots of the running results, to verify the correctness of the following functions included within the UI.~~ ***~~(1 marks)~~***

**~~Show book collection~~**

**~~Add book~~**

**~~Search book by book name~~**

**~~Show list of current members~~**

**~~Add new member~~**

**~~Search member by member id~~**

**~~Borrow book~~**

**~~Return book~~**

**~~Exit~~**

* 1. Demonstration of the solution source code to your tutor during the lab/tutorial class in Week 6. ***(2 marks)***

**NOTE: You must include 5 book objects and 5 member objects within the code as test data within the main method.**

# Task2 Submission –Source Code (Total 5 marks)

Implement and test the source code using a suitable IDE such as Pycharm or Visual Studio code and submit the Python file as a single file named **"Assignment1.py"**.

1. **Code Functionality (2 marks total)**
   1. **Completeness (1 mark)**: Implements all functionalities as specified, including all class methods and user interactions.
   2. **Correctness (1 mark)**: Performs operations correctly, including book and member management, with proper handling of availability status for borrowed and returned books.
2. **Code Quality (1 mark total)**
   1. **Comments and meaningful identifiers (0.5 marks)**: The code is easy to read, with meaningful variables and method names. Comments are used where necessary to explain the purpose of functions and classes.
   2. **Structure and Design (0.5 marks)**: Demonstrates good use of object-oriented programming principles, appropriate use of inheritance, and efficient data structures.
3. **UI Clarity (2 marks total)**
   1. **User Guidance (1 mark)**: Provides clear instructions and prompts to the user for navigating the library system and performing operations.
   2. **All UI options present (1 mark)**: Utilises efficient algorithms for searching and updating data, minimizing unnecessary computations.
   3. **All UI options working correctly (1 mark)**: Each option should work and show relevant and correct output.

# Submission Guidelines

* You will submit **2 files in total** on Moodle
* Submit your files ("**Assignment1\_class\_design.pdf**" and "**Assignment1.py**") to Moodle Assignment 1.
* Note: Email submissions will **NOT** be accepted.
* **Late Submission Penalty**
  + Up to 1 day late: 10% deduction of the original mark.
  + Up to 2 days late: 20% deduction of the original mark.
  + Up to 3 days late: 30% deduction of the original mark.
  + **After 3 days without academic consideration： 0**