Group Assignment 2 - Smart Library System

AI1030 - Python Programming

1 Scenario: "Smart Library System"

Your team has been hired to develop a **digital library system** that stores books and supports efficient search and recommendation. The system should let users:

- Store and manage a collection of books (title, author, genre, year, rating).
- Search for books by keyword or filter (author/genre/year).
- Generate simple recommendations (e.g., top-rated books, or "similar books" by genre).

This project tests your ability to design **Object-Oriented Programming models**, implement **basic and improved algorithms**, and evaluate their **timing performance**.

Figure 1: Library object and features.

2 Tasks

Part A: Core OOP Design

- Implement classes for:
  - `Book` (title, author, genre, year, rating).
  - `Library` (a collection of books, with methods to add/remove/search).
  - `User` (name, borrowed books, history).

Part B: Base Algorithms

- Implement **basic search** for books (linear scan through the collection).
- Implement **basic recommendation** (e.g., list top N rated books).

Part C: Improved Algorithms & Timing

- Implement an **improved search** (e.g., dictionaries, sets, or sorting + binary search).
- Extend recommendations (e.g., recommend books of similar genre with high ratings).
- Compare **base vs. improved algorithms** using the `time` module.

- Measure average search time for 1,000, 10,000, and 50,000 books.
- Present results in a table or chart.

Part D: Reporting Features

- For each run, generate a report including:
  - Number of books in the system.
  - Time taken by base vs. improved search.
  - Best-rated books and recommendations.

Part E: (Optional Extension)

- Add borrowing and returning system with user histories.
- Add recommendation based on a user's borrowed history (e.g., "You borrowed sci-fi, here are other sci-fi books").

3 Deliverables

3.1 Code (40%)

- Well-structured Python code with OOP design.
- Two versions of search and recommendation (base + improved).
- Timing comparison for different dataset sizes.
- At least 5 test cases (adding books, searching, recommendation correctness, timing tests).

3.2 Report (40%)

- Explanation of OOP design.
- Description of base vs. improved search/recommendation.
- Timing results (tables/plots).
- Reflection (300–500 words) on trade-offs and lessons learned.

3.3 Group Presentation (20%)

- Duration: 8–10 minutes.
- All group members must contribute.
- Demonstrate:
  - The system in action (adding books, searching, recommendations).
  - Timing comparison of base vs. improved solutions.
  - Reflections on efficiency.

4 Grading Rubric

Component	**Criteria**	**Points**
Code (40%)	OOP structure, functional library system, base + improved algorithms, timing comparisons, test coverage	40
Report (40%)	Clear explanation, algorithm timing results, reflection on trade-offs	40
Presentation (20%)	All members contribute, demo system + timing results, reflections	20
Extension (2%)	Borrowing/return system, personalized recommendations	2
Total		102

1 Grade Bands

Grade	GPA	Score Range (%)	Performance Description
A+	4.0	95–100	Exceptional work, complete and polished in all aspects; interpreter, report, and presentation exceed expectations.
A	4.0	90–94	Excellent work, very strong across all components with only minor improvements possible.
A-	3.7	85–89	Very good work, complete with small gaps in detail, clarity, or testing.
B+	3.3	80–84	Good work, most requirements met with some issues in implementation or explanation.
B	3.0	75–79	Satisfactory work, functional but with gaps in coverage or clarity.
B-	2.7	70–74	Adequate work, significant issues in one component (e.g., interpreter or report).
C+	2.3	65–69	Passable work, limited testing or reflection, some incomplete elements.
C	2.0	60–64	Barely sufficient, multiple weaknesses across code, report, or presentation.
C-	1.7	55–59	Weak performance, incomplete work with major flaws but some evidence of effort.
D+	1.3	50–54	Very limited work, serious issues across most components, minimal demonstration of understanding.
D	1.0	40–49	Poor work, interpreter largely non-functional or report missing key sections.
F	0.0	0–39	Unsatisfactory, major requirements missing or not attempted.