Presentation (Paste this outline into slides)

Slide 1 — Title

Library Management System (Java, Console)

- Your Name : Boyina Sai Kamalesh
- Internship: VaultOfCode – Final Project

Slide 2 — Introduction

- What it is: console app to manage books & members
- Why: practice methods & data structures (ArrayList, HashMap)

Slide 2 — Requirements / Scope

- Add book, register member
- Borrow/return
- List & search
- Simple validations

Slide 3 — Architecture & Data Structures

- Classes: Book, Member, LibraryService, LibraryApp
- Data: ArrayList<Book>, ArrayList<Member>, Map<Integer, Set<Integer>>
- Explain why you chose them

Slide 4 — Core Methods (Working)

- addBook, addMember
- borrowBook, returnBook
- listBooks, listMembers, searchBooks
- (Show 1–2 code snippets)

Slide 5 — Demo Screenshots

- Add books, members
- Borrow/return
- Search
- List

Slide 6 — Uses

- Small libraries, class assignments
- Demo for data handling and validation
- Base for larger systems (file/DB)

Slide 7 — Advantages

- Simple, modular, beginner-friendly
- Easy to extend (GUI, file save, DB)

Slide 8 — Disadvantages

- No persistent storage (in-memory only)
- No authentication/roles
- No due dates/fines

Slide 9 — Conclusion

- What you learned: methods, collections, validation, separation of logic & UI
- Thank you!

```
import java.util.*;

// ==========================
// Library Management System
// ==========================
public class LibraryApp {

    // ---------------------
    // Book Class
    // ---------------------
    static class Book {
        private static int nextId = 1;
        private int id;
        private String title;
        private String author;
        private int copies;
        private int borrowed;

        public Book(String title, String author, int copies) {
            this.id = nextId++;
```

```
    this.title = title;
    this.author = author;
    this.copies = copies;
    this.borrowed = 0;
}

public int getId() { return id; }
public String getTitle() { return title; }
public String getAuthor() { return author; }
public int getCopies() { return copies; }
public int getBorrowed() { return borrowed; }

public boolean borrow() {
    if (borrowed < copies) {
        borrowed++;
        return true;
    }
    return false;
}

public boolean returnBook() {
    if (borrowed > 0) {
        borrowed--;
        return true;
    }
    return false;
}

@Override
public String toString() {
    return "[" + id + "] " + title + " by " + author +
```

```
            " (Available: " + (copies - borrowed) + "/" + copies + ")";
    }
}

// --------------------
// Member Class
// --------------------
static class Member {
    private static int nextId = 1;
    private int id;
    private String name;

    public Member(String name) {
        this.id = nextId++;
        this.name = name;
    }

    public int getId() { return id; }
    public String getName() { return name; }

    @Override
    public String toString() {
        return "[" + id + "] " + name;
    }
}

// --------------------
// Library Service
// --------------------
static class LibraryService {
    private List<Book> books = new ArrayList<>();
```

```
private List<Member> members = new ArrayList<>();

public Book addBook(String title, String author, int copies) {
   Book b = new Book(title, author, copies);
   books.add(b);
   return b;
}

public Member addMember(String name) {
   Member m = new Member(name);
   members.add(m);
   return m;
}

public List<Book> listBooks() { return books; }
public List<Member> listMembers() { return members; }

public List<Book> searchBooks(String keyword) {
   List<Book> result = new ArrayList<>();
   for (Book b : books) {
      if (b.getTitle().toLowerCase().contains(keyword.toLowerCase()) ||
         b.getAuthor().toLowerCase().contains(keyword.toLowerCase())) {
         result.add(b);
      }
   }
   return result;
}

public Book findBookById(int id) {
   for (Book b : books) if (b.getId() == id) return b;
   return null;
```

```
    }

    public Member findMemberById(int id) {
        for (Member m : members) if (m.getId() == id) return m;
        return null;
    }
}

// --------------------
// Main Menu
// --------------------
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    LibraryService service = new LibraryService();

    while (true) {
        System.out.println("\n===== Library Management System =====");
        System.out.println("1. Add Book");
        System.out.println("2. Add Member");
        System.out.println("3. List Books");
        System.out.println("4. List Members");
        System.out.println("5. Search Books");
        System.out.println("6. Borrow Book");
        System.out.println("7. Return Book");
        System.out.println("0. Exit");
        System.out.print("Enter choice: ");

        int choice = sc.nextInt();
        sc.nextLine(); // consume newline

        switch (choice) {
```

```
            case 1: handleAddBook(sc, service); break;
            case 2: handleAddMember(sc, service); break;
            case 3: handleListBooks(service); break;
            case 4: handleListMembers(service); break;
            case 5: handleSearchBooks(sc, service); break;
            case 6: handleBorrowBook(sc, service); break;
            case 7: handleReturnBook(sc, service); break;
            case 0: System.out.println("Exiting..."); return;
            default: System.out.println("Invalid choice!");
        }
    }
}

// --------------------
// Menu Handlers
// --------------------
private static void handleAddBook(Scanner sc, LibraryService service) {
    System.out.print("Enter book title: ");
    String title = sc.nextLine();
    System.out.print("Enter book author: ");
    String author = sc.nextLine();
    System.out.print("Enter number of copies: ");
    int copies = sc.nextInt();
    sc.nextLine();

    Book b = service.addBook(title, author, copies);
    System.out.println("Added: " + b);
}

private static void handleAddMember(Scanner sc, LibraryService service) {
    System.out.print("Enter member name: ");
```

```
    String name = sc.nextLine();

    Member m = service.addMember(name);
    System.out.println("Added: " + m);
}

private static void handleListBooks(LibraryService service) {
    System.out.println("\nBooks:");
    for (Book b : service.listBooks()) {
        System.out.println(b);
    }
}

private static void handleListMembers(LibraryService service) {
    System.out.println("\nMembers:");
    for (Member m : service.listMembers()) {
        System.out.println(m);
    }
}

private static void handleSearchBooks(Scanner sc, LibraryService service) {
    System.out.print("Enter keyword: ");
    String keyword = sc.nextLine();
    List<Book> found = service.searchBooks(keyword);
    System.out.println("Search Results:");
    for (Book b : found) {
        System.out.println(b);
    }
}

private static void handleBorrowBook(Scanner sc, LibraryService service) {
```

```
    System.out.print("Enter book ID: ");
    int bookId = sc.nextInt();
    System.out.print("Enter member ID: ");
    int memberId = sc.nextInt();
    sc.nextLine();

    Book b = service.findBookById(bookId);
    Member m = service.findMemberById(memberId);

    if (b == null || m == null) {
        System.out.println("Invalid book or member ID!");
        return;
    }
    if (b.borrow()) {
        System.out.println(m.getName() + " borrowed " + b.getTitle());
    } else {
        System.out.println("No available copies for " + b.getTitle());
    }
}

private static void handleReturnBook(Scanner sc, LibraryService service) {
    System.out.print("Enter book ID: ");
    int bookId = sc.nextInt();
    System.out.print("Enter member ID: ");
    int memberId = sc.nextInt();
    sc.nextLine();

    Book b = service.findBookById(bookId);
    Member m = service.findMemberById(memberId);

    if (b == null || m == null) {
```

```
            System.out.println("Invalid book or member ID!");
            return;
        }
        if (b.returnBook()) {
            System.out.println(m.getName() + " returned " + b.getTitle());
        } else {
            System.out.println("This book was not borrowed.");
        }
    }
}
```