

Name = Komal

Roll No = 2401420047

Program = BTech CSE (DS)

Semester = III

Assingement - 4

```
import java.io.*;
```

```
import java.util.*;
```

```
class Book implements Comparable<Books> {
```

```
    int id;
```

```
    String title;
```

```
    String author;
```

```
    String Category;
```

```
    boolean issued;
```

```
    Book (int id, String title, String author, String category,
```

```
        boolean issued) {
```

```
        this.id=id;
```

```
        this.title=title;
```

```
        this.author=author;
```

```
        this.Category=category;
```

```
        this.issued=issued;
```

```
    }
```

```
    void display() {
```

```
        System.out.println("ID: " + id + " | Title: " + title + " | Author: " + author + " | Category: " + Category + " | Issued: " + issued);
```

```
"|category "+Category + " |Issued:" +(issued?"Yes" : "No");
```

```
}
```

```
public int compareTo (Book other) {
```

```
    return this.title.compareToIgnoreCase (other.title);
```

```
}
```

```
String to_Csv() {
```

```
    return id + "," + title + "," + author + "," + category + "," +
```

```
        (issued? "1" : "0");
```

```
}
```

```
Static Book fromCsv (String line) {
```

```
    String[] p = line.split ("", 5);
```

```
    if (p.length < 5) return null;
```

```
    try {
```

```
        int id = Integer.parseInt (p[0]);
```

```
        String title = p[1];
```

```
        String author = p[2];
```

```
        String category = p[3];
```

```
        boolean issued = p[4].trim().equals ("1");
```

```

        return new Book (id, title, author, category, issuedBooks);
    } catch (Exception e) {

```

```

        return null;
    }
}

```

```

class Member {

```

```

    int id;

```

```

    String name;

```

```

    String email;

```

```

    List<Integer> issuedBooks;

```

```

    Member (int id, String name, String email, List<Integer>
        issuedBooks) {

```

```

        this.id = id;

```

```

        this.name = name;

```

```

        this.email = email;

```

```

        this.issuedBooks = (issuedBooks != null) ? issuedBooks

```

```

            : new ArrayList<>();
    }

```

```

    void display() {

```

```

        System.out.print("ID:" + id + "Name:" + name + "|Email:" + email

```

```

            + "|IssuedBooks IDs:" + (issuedBooks.isEmpty() ? "None" :

```

```

                issuedBooks));
    }

```

```
String toCSV() {
```

```
    if (issuedBooks.isEmpty()) {
```

```
        return id + "," + name + "," + email + "," +
```

```
            String.join(",", issuedBooks.stream().map(String::valueOf).toArray(String[]::new));
```

```
    }
```

```
    static Member fromCSV (String line) {
```

```
        String[] p = line.split(",", 4);
```

```
        if (p.length < 4) return null;
```

```
        try {
```

```
            int id = Integer.parseInt(p[0]);
```

```
            String name = p[1];
```

```
            String email = p[2];
```

```
            String listStr = p[3].trim();
```

```
            List<Integer> list = new ArrayList<>();
```

```
            if (!listStr.equals("-")) {
```

```
                for (String s : listStr.split(";")) {
```

```
                    try { list.add(Integer.parseInt(s.trim())); }
                }
```

```
            }
```

```
        }
```

```
        return new Member(id, name, email, list);
```

```
    } catch (Exception e) {
```

```
        return null;
```

```
    }
```

```
public class CityLibraryDigitalManagementSystem {
```

```
    private Map<Integer, Book> books = new HashMap<>();
```

```
    private Map<Integer, Member> member = new HashMap<>();
```

```
    private Map<Integer> Categories = new HashSet<>();
```

```
    private final String BOOKS_FILE = "books.txt";
```

```
    private final String MEMBER_FILE = "member.txt";
```

```
    private Scanner sc = new Scanner(System.in);
```

```
    public static void main (String [] args) {
```

```
        CityLibraryManagementSystemapp = new CityLibraryDigitalManagement
```

```
        app.loadAll();
```

```
        app.runMenu();
```

```
        app.saveAll();
```

```
        app.Sout ("Program finished");
```

```
    }
```

```
    private void loadAll() {
```

```
        loadBooks();
```

```
        loadMembers();
```

```
    }
```

```
    private void loadBooks() {
```

```
        File f = new File(BOOKS_FILE);
```

```
try {
```

```
    if (!f.exists()) f.createNewFile();
```

```
    try (BufferedReader br = new BufferedReader(new FileReader(f));  
        String line;
```

```
        while ((line = br.readLine()) != null) {
```

```
            Book b = Book.fromCSV(line);
```

```
            if (b != null) {
```

```
                books.put(b.id, b);
```

```
                if (b.category != null && !b.category.isEmpty()) {
```

```
                    categories.add(b.category);
```

```
                }
```

```
            }
```

```
        }
```

```
    } catch (Exception e) {
```

```
        Sent("Error loading books" + e.getMessage());
```

```
    }
```

```
}
```

```
private void loadMembers() {
```

```
    File f = new File(MEMBER_FILE);
```

```
    try {
```

```
        if (!f.exists()) f.createNewFile();
```

```
try (BufferedReader br = new BufferedReader(new FileReader(
String line;
while ((line = br.readLine()) != null) {
```

```
    Member m = Member.fromCSV(line);
    if (m != null) member.put(m.id, m);
```

```
}
```

```
}
```

```
} catch (Exception e) {
```

```
    System.out.println("Error loading members: " + e.getMessage());
```

```
}
```

```
}
```

```
private void saveAll() {
```

```
    saveBooks();
```

```
    saveMembers();
```

```
}
```

```
private void saveBooks() {
```

```
    try (BufferedWriter bw = new BufferedWriter(new FileWriter
```

```
        (MEMBER_FILE, false))) {
```

```
        for (Member m : members.values()) {
```

```
            bw.write(m.toCSV());
```

```
            bw.newLine();
```

```
        }
```

```
}
```

Catch (Exception e) {

 Sout("Error savings member: " + e.getMessage());
}

private void runMenu() {

 while (True) {

 Sout("In City Library Menu:");
 Sout("1. Add Book");
 Sout("2. Add Member");
 Sout("3. Issue Book");
 Sout("4. Return Book");
 Sout("5. Search Book");
 Sout("6. Sort Book");
 Sout("7. Show All Book");
 Sout("8. Show All Member");
 Sout("9. Exit");
 Sout("Enter choice:");

 String ch = sc.nextLine().trim();

 switch (ch) {

 case "1": addBook(); break;
 case "2": addMember(); break;
 case "3": issueBook(); break;
 case "4": returnBook(); break;
 case "5": searchBook(); break;
 case "6": sortBookMenu(); break;

```

Case "7" : show AllBooks(); break;
Case "8" : ShowAllMembers(); break;
Case "9" : return ;
default : Sout ("Invalid choice");
}
}
}

```

```

private int nextBook() {

```

```

    if (members.isEmpty()) return 0;
    return Collection.max(bookmembers.keySet()) + 1;
}

```

```

private int nextMemberID() {

```

```

    if (members.isEmpty()) return 201;
    return Collection.max(members.keySet()) + 1;
}

```

```

private void addBook() {

```

```

    int id = nextBookID();

```

```

    Sout ("Title:");
    String title = sc.nextLine().trim();

```

```

    Sout ("Author:");
    String author = sc.nextLine().trim();

```

```
Scout ("Category:");  
String category = sc.nextLine().trim();
```

```
Book b = new Book (id, title, author, category, false);  
books.put (id, b);
```

```
if (!category.isEmpty()) categories.add(category);  
savebooks();
```

```
Scout ("Books added with ID" + id);  
}
```

```
private void returnBook() {
```

```
try {
```

```
Scout ("Member ID:");  
int mid = Integer.parseInt(sc.nextLine().trim());
```

```
Member m = members.get (mid);  
if (m == null) {
```

```
Scout.out.println ("Member not found");  
return;
```

```
}
```

```
Scout ("Books ID:");  
int bid = Integer.parseInt(sc.nextLine().trim());
```

```
book b = books.get (bid);  
if (b == null) {
```

```
Scout ("books not found");
```