Vikram
2401730170
B-Tech CSE(AI/ML)

Date
DELTA Pg No.

Java Assignment-4

Library Management System

Code:-

```
import java.io.*;
import java.util.*;
class Book implements Serializable {
    int BookId;
    String title, author, Category;
    boolean isIssued;

    Book(int id, String t, String a, String c) {
        bookId = id;
        title = t;
        author = a;
        category = c;
        isIssued = false;
    }
    void markIssued() { isIssued = true; }
    void markReturned() { isIssued = false; }

    void show() {
        System.out.println(bookId + " | " + title + " | " + author
        + " | " + category + " | Issued: " + isIssued);
    }
}

public class LibraryManager {
    Map<Integer, Book> books = new HashMap<>();
    Map<Integer, Member> members = new HashMap<>();
    Scanner sc = new Scanner(System.in);
```

```
void load() {
  try {
    books = (Map<Integer, Book>) new ObjectInputStream
    (new FileInputStream("books.dat")).readObject();

    members = (Map<Integer, Member>) new ObjectInput
    Stream(new FileInputStream("members.dat")).read
    Object();
  }
  Catch (Exception e) {
    System.out.println("starting fresh ...");
  }
}
Void save() {
  try {
    new ObjectOutputStream(new FileOutputStream("books.dat"
    )).writeObject(books);
    new ObjectOutputStream(new FileOutputStream("members.dat"
    )).writeObject(members);
  }
  Catch (Exception e) {}

}

Void addBook() {
  System.out.println("Title: ");
  String t = sc.nextLine();
  System.out.println("Author: ");
  String a = sc.nextLine();
  System.out.println("Category: ");
  String c = sc.nextLine();
```

```
int id = books.size() + 101;
books.put(id, new Book(id, t, a, c));
System.out.println("Books added: " + id);
Save();
}

void addMember() {
System.out.println("Book Id: ");
int b = Sc.nextLine();
System.out.println("Member ID: ");
int m = Sc.nextLine();
Sc.nextLine();

if (!books.containsKey(b) || !members.containsKey(m)) {
System.out.println("Invalid IDs!");
return
}

Book bk = books.get(b);
if(bk.isIssued) {
System.out.println("Already Issued!");
return;
}

bk.markIssued();
members.get(m).addBook(b);
System.out.println("Book Issued!");
Save();
}
```

```
void returnBook() {
    System.out.println("Book ID: ");
    int b = sc.nextInt();
    sc.nextLine();

    if(!books.containsKey(b)) {
        System.out.println("Invalid ID!");
        return;
    }
    books.get(b).markReturned();
    for(Member m: members.values()) {
        m.returnBook(b);
    }
    System.out.println("Book Returned!");
    save();
}

void searchBooks() {
    System.out.println("Search: ");
    String key = sc.nextLine().toLowerCase();

    for(Book b: books.values()) {
        if(b.title.toLowerCase().contains(key) ||
           b.author.toLowerCase().contains(key) ||
           b.category.toLowerCase().contains(key)) {
            b.show();
        }
    }
}
```

```
void sortBooks() {
    List<Book> list = new ArrayList<>(books.values());
    System.out.println("1. Sort by Title \n2. Sort by Author");
    int ch = sc.nextInt();
    sc.nextLine();

    if (ch == 1) {
        Collections.sort(list, (x, y) -> x.title.compareToIgnore
        Case(y.title));
    }
    else {
        Collections.sort(list, (x, y) -> x.author.compareToIgnoreCase
        (y.author));
    }
    for(Book b: list)
        b.show();
}

void menu() {
    load();
    while (true) {
        System.out.println("\n 1. Add Book  2. Add Member
        3. Issue  4. Return  5. Search  6. Sort  7. Exit");

        int ch = sc.nextInt();
        sc.nextLine();

        switch (ch) {
            case 1: addBook(); break;
            case 2: addMember(); break;
            case 3: issueBook(); break;
```

```
            case 4: returnBook(); break;
            case 5: searchBooks(); break;
            case 6: sortBooks(); break;
            case 7: save();
            System.out.println("Good Bye!");
            return;
            default: System.out.println("Invalid choice!");
        }
    }
}

public static void main(String[] args) {

    new LibraryManager().menu();
    }
}
```