Date / /
Page No.

Java Assignment - IV

Krish Jaiska, 2401730188

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

class Book implements Comparable<Book> {
    int bookId;
    String title;
    String author;
    String category;
    boolean isIssued;
    static int baseBookId = 100;
    Book(String title, String author, String category) {
        this.bookId = ++baseBookId;
        this.title = title;
        this.author = author;
        this.category = category;
        this.isIssued = false;
    }
    Book(int id, String title, String author, String category,
        boolean isIssued) {
        this.bookId = id;
        this.title = title
        this.author = author;
        this.category = category;
        this.isIssued = false;
    }
    Book(int id, String title, String author, String category,
        boolean is
        if(id > baseBookId) {
            baseBookId = id;
        }
}
```

```
void displayBookDetails(){
    System.out.println("ID:" + bookId +
                       "\nTitle:" + title +
                       "\nAuthor:" + author +
                       "\nCategory:" + category +
                       "\nAvailable:" + (!isIssued));
}
public int compareTo(Book other){ return this.title.com
    -compareToIgnoreCase(other.title); }
}class Member{
    int memberId;
    String name;
    String email;
    ArrayList<Integer> issuedBooks = new ArrayList<>();
    static int baseMemberId = 200;
    Member(String name, String email){
        this.memberId = ++baseMemberId;
        this.name = name;
        this.email = email;
    }
    Member(int id, String name, String email){
        this.memberId = id;
        this.name = name;
        this.email = email;
        if(id > baseMemberId){
            baseMemberId = id;
        }
    }
    void displayMemberDetails(){
        System.out.println("ID:" + memberId +
                           "\nName:" + name +
                           "\nEmail:" + email +
        "\nIssued Books ID:" + issuedBooks.toString());
    }
```

Date / /
Page No.

```
void addIssueBook(int bookId) { issuedBooks.add(bookId); }
void returnIssuedBook(int bookId) { issuedBooks.remove(
    Integer.valueOf(bookId)); }
} class LibraryManager {
    HashMap<Integer, Book> books = new HashMap<>();
    HashMap<Integer, Member> members = new HashMap<>();
    Scanner sc = new Scanner(System.in);
    String BOOK_FILE = "Books.txt";
    String MEMBER_FILE = "members.txt";
    void saveToFile() {
        try (BufferedWriter bwBook = new BufferedWriter(
            new FileWriter(BOOK_FILE));
            BufferedWriter bwMember = new BufferedWriter(
            new FileWriter(MEMBER_FILE))) {
            for (Book b : books.values()) {
                bwBook.write(String.valueOf(b.bookId));
                bwBook.newLine();
                bwBook.write(b.title); bwBook.newLine();
                bwBook.write(b.author); bwBook.newLine();
                bwBook.write(b.category); bwBook.newLine();
                bwBook.write(String.valueOf(b.isIssued));
                bwBook.newLine();
            } for (Member m : members.values()) {
                bwMember.write(String.valueOf(m.memberId));
                bwMember.newLine();
                bwMember.write(m.name); bwMember.newLine();
                bwMember.write(m.email); bwMember.newLine();
                String issued = "";
                for (int id : m.issuedBooks) {
                    issued += id + ",";
                }
```

Date / /
Page No.

```
            if (!issued.isEmpty()){
                issued = issued.substring(0, issued.length()-1);
            }
            bwMember.write(issued);
            bwMember.newLine();}
        System.out.println("Data saved successfully.");
    } catch (IOException e){
        System.out.println("Error saving data.");
    }
}

void addBook(){
    System.out.println("Enter title:");
    String title = sc.nextLine();
    System.out.println("Enter Author:");
    String author = sc.nextline();
    System.out.println("Enter Category:");
    String cat = sc.nextline();
    Book newBook = new Book(title, author, cat);
    books.put(newBook.bookId, newBook);
    System.out.println("Book added:" + newBook.
        bookId); saveToFile();
}
void addMember(){
    System.out.println("Enter Name:");
    String name = sc.nextline();
    System.out.println("Enter Email:");
    String email = sc.nextline();
    Member newMember = new Member(name,
    email);
    members.put(newMember.memberId, newMember);
    System.out.println("Member added:"+memberId);
    saveToFile();
}
```

Date / /
Page No.

```
void issuedBook(){
        System.out.println("Enter book ID: ");
        int bid = sc.nextInt();
        System.out.println("Enter Member ID: ");
        int mid = sc.nextInt();
        sc.nextLine();
        if (books.containsKey(bid) && members.containsKey
        (mid)){
            Book b = book.get(bid);
            Member m = members.get(mid);
            if(!b.isIssued){
                b.isIssued = true;
                m.addIssuedBook(bid);
                System.out.println("Book issued
                    successfully.");
                saveToFile();
            }else{
                System.out.println("Book is already
                    issued.");
            }
        }else{
            System.out.println("Invalid IDs");
        }
    }
void returnBook(){
        System.out.print("Enter book ID to return");
        int bid = sc.nextInt();
        System.out.print("Enter Member ID returning: ");
        int mid = sc.nextInt();
        sc.nextLine();
```

Date / /

Page No.

```
if (books.containsKey(bid) && members.containsKey(mid)){
    Book b = books.get(bid);
    Member m = members.get(mid);
    if (b.isIssued && m.issuedBooks.contains(bid)){
        b.isIssued = false;
        m.returnIssuedBook(bid);
        System.out.println("Books return success");
        saveToFile();
    } else {
        System.out.println("This member does
            not have this book.");
    }
} else {
    System.out.println("This member does not have
        this book.");
}
}
void searchBooks(){
    System.out.print("Enter search term (Title/Author
        /Category):");
    String term = sc.nextLine().toLowerCase();
    boolean found = false;
    for(Book b : books.values()){
        if(b.title.toLowerCase().contains(term) ||
            b.author.toLowerCase().contains(term) ||
            b.category.toLowerCase().contains(term)){
            b.displayBookDetails();
            found = true;
        }
    } if(!found){
        System.out.println("No book found");
    }
}
```

Date / /
Page No.

```
void sortBooks(){
    System.out.println("1. Sort by Title \n2. Sort by
        Author");
    int option = sc.nextInt();
    sc.nextLine();
    List<Book> bookList = new ArrayList<>(books.
     values());
    if (option == 1){
        Collations.sort(bookList);
    } else {
        collections.sort(booklist, new AuthorComparator
         ());
    }
    for (Book b : bookList){
        b.displayBookDetails();
    }
}
void menu(){
    loadFromFile();
    while(true){
        System.out.println("Menu"+
                           "\n1) Add Book"+
                           "\n2) Add Member"+
                           "\n3) Issue Book"+
                           "\n4) Return Books"+
                           "\n5) Search Books"+
                           "\n6) Sort Books"+
                           "\n7) Exit"+
                           "\nChoose your option:");

    try{
        int option = nextInt();
        sc.nextLine();
```

Date / /
Page No.

```
switch (option){
    case 1:
        addBook();
        break;
    case 2:
        addMember();
        break;
    case 3:
        issueBook();
    ~~case 4:~~ break;
    case 4:
        returnBook();
        break;
    case 5:
        searchBooks();
        break;
    case 6:
        sortBooks();
        break;
    case 7:
        saveToFile();
        System.out.println("Exiting program");
        return;
    default:
        System.out.println("Invalid option");
}

} catch (InputMismatchException e){
    System.out.println("Invalid input! Please
        enter a number.");
    sc.nextLine();
} catch (Exception e) {
```

Date / /

Page No.

```
                System.out.println("Error: "+e.getMessage());
            }
        }
    }
    public static void main(String[] args) {
        LibraryManager ai = new LibraryManager();
        ai.menu();
    }
}
```