Name - Jahanvi
Rollno - 2401420026
BTech CSE Data Science

Java Assignment - 4 →

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

static class Book implements Comparable<Book>{
int id;
String name;
String author;
String category;
boolean issued;

Book(int id, String name, String author,
String category, boolean issued){
    this.id = id;
    this.name = name;
    this.author = author;
    this.category = category;
    this.issued = issued;
}
  void show(){
      Sout("Book ID : " + id + name +
          author + category + issued");
}
public int compareTo(Book b){
  return this.name.compareToIgnoreCase
          (b.name);
}
}
}
```

__/__/____

```
static class Member {
int id;
String name;
String email;
List<Integer> borrowedBooks = new ArrayList<>();
    Member(int id, String name, String
email) {
        this.id = id;
        this.name = name;
        this.email = email;
  }
void show() {
  Sout("Member ID: " + id + name +
  " Email:" + email);
  }
}

static Map<Integer, Book> bookList = new
    HashMap<>();

static Map<Integer, Member> memberList=
new HashMap<>();

  static File bookFile = new File("books.txt");
  static File memberFile = new File("members.txt");

  static void loadData(){
    try {
      if(bookFile.exists()){
        BufferReader br = new BufferedReader(
new FileReader(bookFile));
        String line;
        while((line = br.readline()) != null) {
```

```
String line;
while ((line = br.readline()) != null ){

String [] data = line.split(",");
        int id = Integer.parseInt(data[0]);
        bookList.put(id, new Book(id,
data[1], data[2], data[3], Boolean.
Boolean.parseBoolean(data[4])));
        }
        br.close();
    }
    if(memberFile.exists()){
        BufferedReader br = new BufferedReader
(new FileReader(memberFile));
    String line;
    while((line = br.readline()) != null){
        String[] data = line.split(",");
    int id = Integer.parseInt(data[0]);
        Member m = new Member(id,
data[1], data[2]);
            for(int i=3; i < data.length; i++){
m.borrowedBooks.add(Integer.parseInt(data[i]));
        }
        memberList.put(id, m);
    }
    br.close();
}
catch(Exception e){
    Sout("Error" + e.getMessage());
    }
}
```

```
static void saveData() {
  try {
     BufferedWriter bw = new BufferedWriter
(new FileWriter(bookFile));
   for (Book b: bookList.values()) {
       bw.write(b.id + "," + b.name + "," +
b.author + "," + b.category + "," +
    b.issued);
        bw.newline();
    }
    bw.close();
BufferedWriter (new FileWriter(memberfile));
    for(Member m: memberlist.values()){
     bw2.write(m.id + "," + m.name +
"," + m.email);
        for (int bookID: m.borrowedBooks)
     bw2.write(",," + bookID);
     bw2.newline();
   }
   bw2.close();
  }
  catch (Exception e) {
   Sout("Error" + e.getMessage());
    }
  }

static void addBook(Scanner sc) {
 Sout("Enter Book ID: ");
int id = sc.nextInt; sc.nextLine();
 Sout("Enter Book Name: ");
String name = sc.nextLine();
Sout("Enter Author Name: ");
```

```
String author = sc.nextLine();
Sout ("Enter Category :");
Sout String category = sc.nextLine();

if (bookList.containsKey(id)){
Sout ("Book ID already exists!");
    return ;
}
booklist.put (id, new Book (id, name, author,
category, false));
   saveData();
Sout ("Book added successfully!");
        }
static void borrowBook (Scanner sc){
Sout ("Enter Book ID:");
int bid = sc.nextInt();
Sout ("Enter MemberID:");
int mid = sc.nextInt();

if (! booklist.containsKey(bid)){
    Sout ("Book ID not found");
        return;
}
b.issued = true;
if (! memberlist.containsKey(mid)){
    Sout ("Member ID not found");
        return;
}
Book b = booklist.get(bid);
Member m = memberlist.get(mid);

    if (b.issued){
```

```
m.borrowedBooks.add(bid);
saveData();
Sout("Book borrowed successfully");
}

static void returnBook(Scanner sc){
Sout("Enter Book ID: ");
int bid = sc.nextInt();
Sout("Enter Member ID: ");
int mid = sc.nextInt();

if(!bookList.~~get(bid~~.containsKey(bid) || !
memberList.containsKey(mid)){
    Sout("Invalid Book or member ID");
    return;
}

Book b = bookList.get(bid);
Member m = memberList.get(mid);

if(!b.issued || !m.borrowedBooks.contains(bid)){
Sout("This book was not issued by this member");
    return;
}
    b.issued = false;
m.borrowedBooks.remove(Integer.valueOf(bid));
    saveData();
Sout("Book returned successfully");
    }
static void searchBook(Scanner sc){
 sc.nextLine();
Sout("Enter keyword to search");
```

```
String key= sc.nextLine().toLowerCase();
    boolean found = false;
    for (Book b: booklist.values()){
      if
(b.name.toLowerCase().contains(key)||
b.author.toLowerCase().contains(key)||
b.category.toLowerCase().contains(key)){
            b.show();
            found = true;
    }
  }
if (!found) Sout ("No matching books found!");
  }
static void sortBooks(){
  List<Book> list = new
ArrayList<>(booklist.values());
Collections.sort(list);
Sout(" Sorted Books");
for (Book b: list) b.show();
  }
psvm (String [] a){
Scanner sc= new Scanner (System.in);
loadData();
while(true){
Sout(" Welcome to Library");
Sout (" 1. Add Book ");
Sout (" 2. Add Member ");
Sout ("3. Borrow Book ");
Sout (" 4. Return Book");
Sout (" 5. Search Book ");
Sout (" 6. Show Sorted Book");
Sout (" 7. Exit");
```

```
Sout (" Choose ");
int choice;
if (sc.hasNextInt()) {
    choice = sc.nextInt();
}
else {
sc.nextLine();
Sout ("Please enter a valid number!");
    continue;
}
switch (choice) {
case 1: addBook(sc); break;
Case 2: addMember(sc);
break;
Case 3: borrowBook(sc);
break;
case 4: returnBook(sc);
break;
Case 5: searchBook(sc);
break;
Case 6: sortBooks();
break;
Case 7:
saveData();
sout ("Goodbye");
    return;
default:
    Sout ("Invalid Choice. Try again");
}
}
}
}
```