Assignment-4

```
import java.io.*;
import java.util.*;
public class LibrarySystem {
static class Book implements Comparable <Book>{
        Integer bookId;
        String title;
        String author;
        String category;
        boolean isIssued;
public Book (Integer bookId, String title, String author,
        String category){
                this.bookId = bookId;
                this.title = title;
                this.author = author;
                this.category = category;
                this.isIssued = false;
        }
public void markAsIssued(){
        this.isIssued = true; }
public void markAsReturned(){
        this.isIssued = false; }
public void displayBookDetails (){
        sout("Book ID :" + bookId);
        sout("Title :" + title);
        sout ("Author :" + author);
        sout ("Category :" + category);
        sout ("Issued :" + (isIssued) "Yes" : "No
        sout("-----");
        }
public int compareTo(Book b){
```

```
return
this.title.compareToIgnoreCase(b.title);
}
public String toString() {
    return bookId + "," + title + "," + author + "," +
    category + "," + isIssued;
}
public static Book fromString(String line) {
    String[] p = line.split(",");
    Book book = new Book(Integer.parseInt(p[0]), p[1],
    p[2], p[3]);
    if (Boolean.parseBoolean(p[4])) book.markAsIssued();
    return book;
}
}
static class Member {
    Integer memberId;
    String name;
    String email;
List<Integer> issuedBooks = new ArrayList<>();
public Member(Integer memberId, String name, Strin
email) {
    this.memberID = memberId;
    this.name = name;
    this.email = email;
}
public void addIssuedBook(int bookID) { issuedBoo
    add(bookID); }
public void returnIssuedBook(int bookID){
    issuedBooks.remove(Integer.valueOf(bookID)); }
public void displayMemberDetails(){
    Sout("Member ID: " + memberID);
```

```
sout("Name:" + name);
sout("email:" + email);
sout("Issued Books:" + issuedBooks);
public String toString(){
return memberID + "," + name + "," + email + "," +
issuedBooks.toString();
}
public static Member fromString(String line){
String[] p = line.split(",");
Member m = new Member(Integer.parseInt(p[0]),
p[1], p[2]);
if(p.length > 3 || p[3].contains("["){
String books = p[3].replace("[", "").replace
("]", "");
if(!books.isEmpty()){
for(String s: books.split(" "))
m.addIssuedBook(Integer.parseInt(s));
}
}
return m;
}
}
static class LibraryManager{
Map<Integer Book> books = new HashMap<>();
Map<Integer Member> members = new HashMap<>();
file bookFile = new file("books.text");
file memberfile = new file("members.int");
public void loadfromfile(){
try{
if(!bookfile.exists())
bookfile.createNewfile();
membersfile.createNewfile();
```

```
public void saveToFile(){
  try{
  BufferedWriter bw = new BufferedWriter(new
  FileWriter(bookFile));
    for(Book b: books.values()){
  bw.write(b.toString());
  bw.newLine();
  }
  bw.close();
  BufferedWriter bw2 = new BufferedWriter(n
    FileWriter(memberFile));
    for(Member m: members.value()){
  bw2.write(m.toString());
  bw2.newLine();
  }
  bw2.close();
  } catch(Exception e){
    e.printStackTrace();
  }
}
public void addBook(Scanner sc){
  System.out("Enter Book ID");
  int id = sc.nextInt();
  sc.nextLine();
  System.out("Enter Title");
  String title = sc.nextLine(
  System.out("Enter author");
  String author = sc.nextLine();
  System.out("Enter Category:");
  String Category = sc.nextLine();
```

```
Book b = new Book(id, title, author, category);
    books.put(id, b);
    saveToFile();
    System.out.println("Book added successfully!");
}
public void addMember(Scanner sc){
    System.out.println("Enter member ID");
    int id = sc.nextInt();
    sc.nextLine()
    System.out.println("Enter name");
    String name = sc.nextLine();
    System.out.println("enter email");
    String email = sc.nextLine();
    Member m = new Member(id, name, email)
    members.put(id, m);
    saveToFile();
    System.out.println("Member added succesfully!");
public void issueBook(Scanner sc){
    System.out.println("Enter BookID");
    int bookId = sc.nextInt();
    System.out.println("Enter memberId");
    int memberId = sc.nextInt();
    if(!books.containsKey(bookId)){
    System.out.println("Book not found!");
        return;
    }
        if(!members.containsKey(memberI
        System.out.println("Member not found!");
            return;
        }
```

```
Book b = books.get(bookId);
Member m = members.get(memberId);
if (b.isIssued()){
    System.out.println("Book already issued!");
    return;
}
b.markAsIssued();
m.addIssuedBook(bookId);
saveToFile();
System.out.println("Book issued successfully!");
public void returnBook(Scanner sc){
System.out.println("Enter Book Id");
int bookId = sc.nextInt();
System.out.println("Enter Member Id");
int memberId = sc.nextInt();
if(!books.containsKey(bookId) ||
!members.containsKey(memberId)){
System.out.println("Invalid details!");
    return;
}
public void searchBooks(Scanner sc){
    sc.nextLine();
    System.out.println("search by title/author/category");
    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.displayBookDetails();
}
}
}
```

```
public static void main(String[] args) {
    LibraryManager lm = new LibraryManager();
    lm.loadFromFile();
    Scanner sc = new Scanner(System.in);
    while(true) {
    Sout("\n ===== City Library Digital Management
        System");
    Sout("1. add book");
    Sout("2. add member");
    Sout("3. Issue Book");
    Sout("4. Return Book");
    Sout("5. SearchBook");
    Sout("6. Sort books");
    Sout("7. Quits");
    Sout("enter your choice:");
    int ch = sc.nextInt();
    switch(ch) {
        case 1:
lm.addBook(sc); break;
                case 2:
lm. addMember(sc); break;
                case 3:
lm. issueBook(sc); break;
                case 4:
lm. returnBook(sc); break;
                case 5:
lm. searchBook(sc); break;
                case 6:
lm. sortBooks(); break;
                case 7:
lm. saveToFile();
    Sout("exiting");
            return;
```

```
BufferedReader br = new BufferedReader (new FileReader
                    (bookFile));
String line;
while( line = br.readline ()) != null){
book b = Book.fromString (line);
books.put (b.bookId, b);
}
br.close ();
BufferedReader br2 = new BufferedReader(ne
  fileReader (memberFile));
while ((line = br2.readLine()) != null)
member m = member.fromString (Line);
members.put(m.memberID, m);
}
br2.close ();
}
catch (Exception){
    e.printStackTrace ();
}
}
```

```
default:
cout<<("Invalid Choice!");
}
}
}
```