Name : Mali Anjali Prakash

Roll no: 27

Assigment no 4

Assignment title: Design a system for maintaining information of any real life problem. Implement inheritance concept in it

// Base class representing a generic Book

class Book {

    private String title;

    private String author;

    private int pages;

    // Constructor to initialize book details

    public Book(String title, String author, int pages) {

        this.title = title;

        this.author = author;

        this.pages = pages;

    }

    // Getters and Setters for attributes

    public String getTitle() {

        return title;

    }

    public String getAuthor() {

        return author;

    }

    public int getPages() {

        return pages;

    }

    // Method to display book information (Abstraction)

    public void displayInfo() {

        System.out.println("Title: " + title);

        System.out.println("Author: " + author);

        System.out.println("Pages: " + pages);

    }

}

// Derived class for Regular Books (Inherits from Book)

class RegularBook extends Book {

    private String shelfLocation; // Specific to regular books

    // Constructor for RegularBook

    public RegularBook(String title, String author, int pages, String shelfLocation) {

        super(title, author, pages); // Calling the base class constructor

        this.shelfLocation = shelfLocation;

    }

    // Getter and Setter for shelfLocation

    public String getShelfLocation() {

        return shelfLocation;

    }

    // Overriding displayInfo() to include shelfLocation for regular books

    @Override

    public void displayInfo() {

        super.displayInfo(); // Display common book info

        System.out.println("Shelf Location: " + shelfLocation);

    }

}

// Derived class for E-books (Inherits from Book)

class EBook extends Book {

    private String fileFormat; // Specific to e-books

    // Constructor for EBook

    public EBook(String title, String author, int pages, String fileFormat) {

        super(title, author, pages); // Calling the base class constructor

        this.fileFormat = fileFormat;

    }

    // Getter and Setter for fileFormat

    public String getFileFormat() {

        return fileFormat;

    }

    // Overriding displayInfo() to include fileFormat for e-books

    @Override

    public void displayInfo() {

        super.displayInfo(); // Display common book info

        System.out.println("File Format: " + fileFormat);

    }

}

// Class representing a Library Member

class LibraryMember {

    private String name;

    private String memberId;

    // Constructor to initialize member details

    public LibraryMember(String name, String memberId) {

        this.name = name;

        this.memberId = memberId;

    }

    // Getter and Setter for member details

    public String getName() {

        return name;

    }

    public String getMemberId() {

        return memberId;

    }

    // Method to display member details

    public void displayMemberInfo() {

        System.out.println("Member Name: " + name);

        System.out.println("Member ID: " + memberId);

    }

}

// Class representing a Library system to manage books and members

class Library {

    private Book[] books;

    private LibraryMember[] members;

    private int bookCount = 0;

    private int memberCount = 0;

    // Constructor to initialize library with a certain size

    public Library(int bookCapacity, int memberCapacity) {

        books = new Book[bookCapacity];

        members = new LibraryMember[memberCapacity];

    }

    // Method to add a book to the library

    public void addBook(Book book) {

        if (bookCount < books.length) {

            books[bookCount++] = book;

        } else {

            System.out.println("Library is full, cannot add more books.");

        }

    }

    // Method to add a member to the library

    public void addMember(LibraryMember member) {

        if (memberCount < members.length) {

            members[memberCount++] = member;

        } else {

            System.out.println("Library membership is full.");

        }

    }

    // Method to display all books in the library

    public void displayAllBooks() {

        System.out.println("Books in Library:");

        for (int i = 0; i < bookCount; i++) {

            books[i].displayInfo();

            System.out.println("------------------------------");

        }

    }

    // Method to display all library members

    public void displayAllMembers() {

        System.out.println("Library Members:");

        for (int i = 0; i < memberCount; i++) {

            members[i].displayMemberInfo();

            System.out.println("------------------------------");

        }

    }

}

// Main class to test the Library Management System

public class inhert{

    public static void main(String[] args) {

        // Create a Library system with capacity for 5 books and 3 members

        Library library = new Library(5, 3);

        // Create books and members

        Book book1 = new RegularBook("The Great Gatsby", "F. Scott Fitzgerald", 180, "Shelf A1");

        Book book2 = new EBook("Digital Fortress", "Dan Brown", 340, "PDF");

        LibraryMember member1 = new LibraryMember("Alice", "M001");

        LibraryMember member2 = new LibraryMember("Bob", "M002");

        // Add books and members to the library

        library.addBook(book1);

        library.addBook(book2);

        library.addMember(member1);

        library.addMember(member2);

        // Display all books in the library

        library.displayAllBooks();

        // Display all members of the library

        library.displayAllMembers();

    }

}

