## OOPs Programming

1.Design a Java program that uses OOP principles to model the Book. Create two classes: Book and Library. The Book class should have attributes such as bookID, title, author, and is Available. The Library class should include an array to store book objects.

Provide methods to add books, remove book search books (using id)and display books.

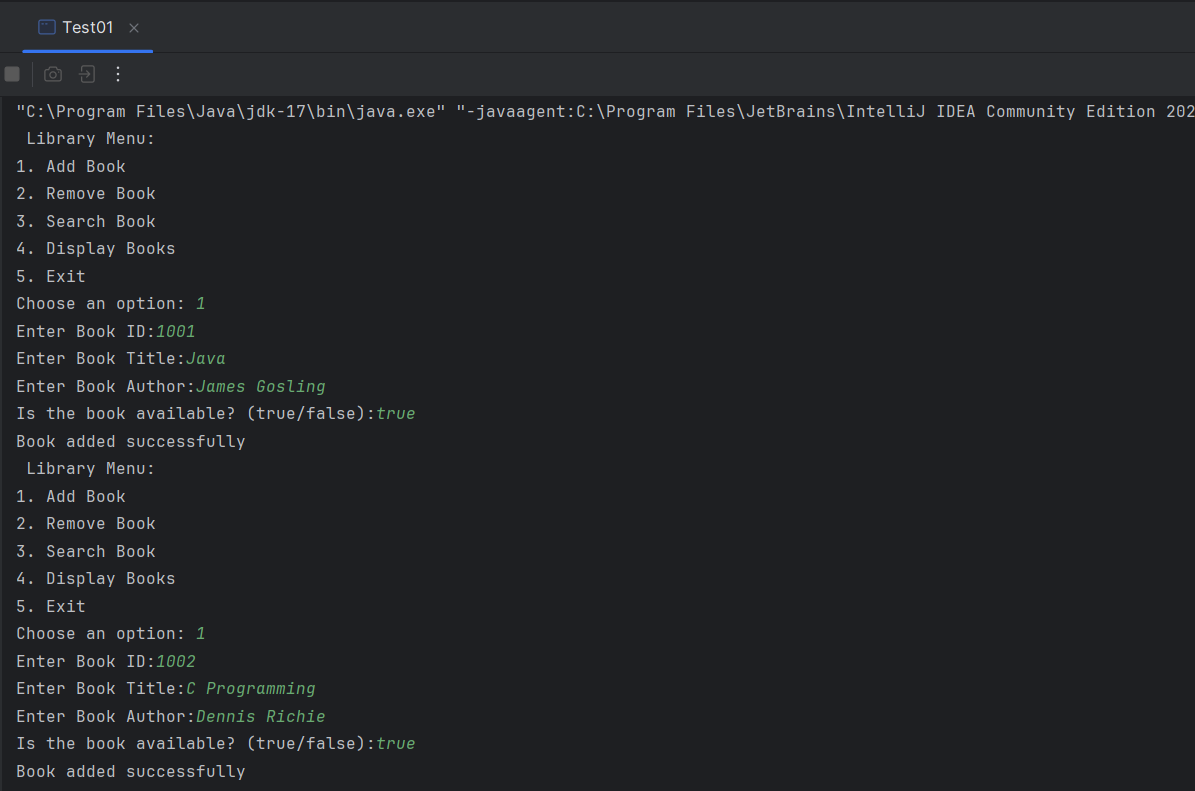
Write a Java program that demonstrates the use of these classes and methods by allowing the user to interact with the library system.

public class Book {  
 // Private fields to store the book details  
 private int bookId;  
 private String title;  
 private String author;  
 private boolean isAvailable;  
 // Default constructor  
 public Book() {  
 }  
 // Parameterized constructor initializes book object  
 public Book(int bookId, String title, String author, boolean isAvailable) {  
 this.bookId = bookId;  
 this.title = title;  
 this.author = author;  
 this.isAvailable = isAvailable;  
 }  
 // Getter for bookId  
 public int getBookId() {  
 return bookId;  
 }  
 // Setter for bookId  
 public void setBookId(int bookId) {  
 this.bookId = bookId;  
 }  
 // Getter for title  
 public String getTitle() {  
 return title;  
 }  
 // Setter for title  
 public void setTitle(String title) {  
 this.title = title;  
 }  
 // Getter for author  
 public String getAuthor() {  
 return author;  
 }  
 // Setter for author  
 public void setAuthor(String author) {  
 this.author = author;  
 }  
 // Getter for isAvailable  
 public boolean isAvailable() {  
 return isAvailable;  
 }  
 // Setter for isAvailable  
 public void setAvailable(boolean available) {  
 isAvailable = available;  
 }  
 // Overrides the toString method()  
 public String toString(){  
 return " BookID: "+getBookId()+" Title: "+getTitle()+" Author: "+getAuthor()+" Available:"+(isAvailable ? " yes " : " no ");  
 }  
}

public class Library {  
 // Array to store Book objects  
 private Book[] books;  
 private int count;  
 // Constructor to initialize the library  
 public Library() {  
 this.books = new Book[5]; // Initializing books array with size 5  
 count=0;  
 }  
 // Method to add new book  
 public void addBooks(int bookId, String title, String author, boolean isAvailable) {  
 if(count>=books.length){ // checks whether library is full or not  
 System.*out*.println("Library is full");  
 }else{  
 // Create new Book object and add it to books array.  
 books[count] =new Book(bookId, title, author, isAvailable);  
 count++;  
 System.*out*.println("Book added successfully");  
 }  
 }  
 // Method to remove book  
 public void removeBook(int bookID){  
 // Loop to find the book with the givenID to remove it  
 for (int i = 0; i < books.length; i++) {  
 if (books[i].getBookId() == bookID) {  
 for (int j = i; j < count - 1; j++) {  
 books[j] = books[j + 1];  
 }  
 books[count - 1] = null;  
 count--;  
 System.*out*.println("Book removed successfully");  
 return;  
 }  
 }  
 }  
 // Method to search book  
 public void searchBooks(int bookID){  
 // Loop to find the book with the given ID.  
 for (int i=0;i<count;i++){  
 if(this.books[i].getBookId()==bookID){  
 //this.books[i].toString();  
 // if book is found print its details  
 System.*out*.println(this.books[i].toString());  
 return;  
 }  
 }  
 System.*out*.println("Book not found");  
 }  
 // Method to display books  
 public void displayBooks(){  
 if(count==0) {  
 // no books in the library  
 System.*out*.println("Library is empty");  
 }else{  
 for (Book bookDetails : books) {  
 if (bookDetails != null) {  
 System.*out*.println(bookDetails.toString());  
 }  
 }  
 }  
 }  
  
}

import java.util.Scanner;  
  
public class Test01 { // class with main method  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*); // Scanner class object to get user input  
 // Create an instance of the Library class  
 Library library=new Library();  
 // Infinite loop to display menu and take user input until they choose to exit  
 while(true) {  
 // Display menu options to user  
 System.*out*.println(" Library Menu:");  
 System.*out*.println("1. Add Book");  
 System.*out*.println("2. Remove Book");  
 System.*out*.println("3. Search Book");  
 System.*out*.println("4. Display Books");  
 System.*out*.println("5. Exit");  
 System.*out*.print("Choose an option: ");  
 int choice = sc.nextInt(); // Read user's choice  
 sc.nextLine();  
 // Perform action based on the user's choice  
 switch (choice) {  
 case 1: // Option to add book  
 System.*out*.println("Enter Book ID:");  
 int bookId = sc.nextInt();  
 sc.nextLine();  
 System.*out*.println("Enter Book Title:");  
 String title = sc.nextLine();  
 System.*out*.println("Enter Book Author:");  
 String author = sc.nextLine();  
 System.*out*.println("Is the book available? (true/false):");  
 boolean isAvailable = sc.nextBoolean();  
 library.addBooks(bookId, title, author, isAvailable); // calls addBooks method  
 break;  
 case 2: // Option to remove book  
 System.*out*.println("Enter Book ID to remove:");  
 int removeId = sc.nextInt();  
 library.removeBook(removeId); // calls removeBooks method  
 break;  
 case 3: // Option to search book  
 System.*out*.println("Enter Book ID to search:");  
 int searchId = sc.nextInt();  
 library.searchBooks(searchId); // calls searchBooks method  
 break;  
 case 4: // Option to display book  
 library.displayBooks(); // calls displayBooks method  
 break;  
 case 5: // Option to exit  
 System.*out*.println("Exit..");  
 sc.close(); // close Scanner class object  
 return;  
 default: // handles invalid choices  
 System.*out*.println("Invalid choice.");  
 }  
 }  
 }  
}

Output:



A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

2.Create Interface Taxable with members sales Tax-7% and incomeTax-10.5%. create abstract method calcTax().

a. Create class Employee(empId,name, salary) and implement Taxable to calculate income Tax on yearly salary.

b. Create class Product(pid,price,quantity) and implement Taxable to calculate sales Tax on unit price of product.

c. Create class for main method(Say DriverMain), accept employee information and a product information from user and print income tax and sales tax respectively.

public interface Taxable {  
 double *salesTax*=0.07; // Sales tax rate as constant.  
 double *incomeTax*=0.105; // Income tax rate as constant.  
 // Abstract method to calculate tax, will be implemented by classes  
 void calcTax();  
  
}

// Employee class implements Taxable interface  
public class Employee implements Taxable{  
 // Private fields to store employee details  
 private int empId;  
 private String name;  
 private double salary;  
  
 // Constructor to initialize the employee object  
 public Employee(int empId, String name, double salary){  
 this.empId=empId;  
 this.name=name;  
 this.salary=salary;  
  
 }  
 @Override  
 // Overriding calcTax method from Taxable interface  
 public void calcTax() { // calculate IncomeTax on yearly salary  
 double yearlySalary=salary\*12;  
 double incomeTaxAmount=yearlySalary\**incomeTax*;  
 System.*out*.println("Employee ID: "+empId+" Name: "+name+" Income Tax: "+incomeTaxAmount);  
 }  
}

// Product class implements Taxable interface  
public class Product implements Taxable {  
 // Private fields to store product details  
 private int pid;  
 private double price;  
 private int quantity;  
 // Constructor to initialize product  
 public Product(int pid, double price, int quantity){  
 this.pid=pid;  
 this.price=price;  
 this.quantity=quantity;  
 }  
  
 @Override  
 //Overriding calcTax method from Taxable interface  
 public void calcTax() {  
 // Calculate sales tax for single unit of the product.  
 double salesTaxUnitProduct= price \* *salesTax*;  
 System.*out*.println("Pid: "+pid+ " Price: "+price+ " SalesTax: "+salesTaxUnitProduct);  
  
 }  
}

import java.util.Scanner;  
  
public class Test02 { // class with main method  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*); // Scanner class object to get user input  
 // Accepting multiple employees  
 System.*out*.print("Enter no. of employees: ");  
 int employeeCount = sc.nextInt(); // no. of employees  
 //Creates an array of Employee objects with a size of employeeCount.  
 Employee[] employees = new Employee[employeeCount];  
 // Loop to read input  
 for (int i = 0; i < employeeCount; i++) {  
 System.*out*.println("Enter employee"+(i+1)+" details ");  
 System.*out*.print("Enter employee ID: ");  
 int empId = sc.nextInt();  
 sc.nextLine();  
 System.*out*.print("Enter employee name: ");  
 String name = sc.nextLine();  
 System.*out*.print("Enter employee monthly salary: ");  
 double salary = sc.nextDouble();  
 // Creating Employee object and store it in the array  
 employees[i] = new Employee(empId, name, salary);  
 }  
 sc.nextLine();  
 sc.nextLine();  
  
 // Accepting multiple products  
 System.*out*.print("Enter the number of products: ");  
 int productCount = sc.nextInt(); // no. of products  
 //Creates an array of Product objects with a size of productCount.  
 Product[] products = new Product[productCount];  
 // Loop to read input  
 for (int i = 0; i < productCount; i++) {  
 System.*out*.println("Enter product"+(i+1)+" details");  
 System.*out*.print("Enter Product ID: ");  
 int pid = sc.nextInt();  
 System.*out*.print("Enter Product Price: ");  
 double price = sc.nextDouble();  
 System.*out*.print("Enter Product Quantity: ");  
 int quantity = sc.nextInt();  
 // Creating Product object and store it in the array  
 products[i] = new Product(pid, price, quantity);  
 }  
  
 // Display income tax for each employee  
 System.*out*.println("Income Tax for Employees:");  
 for (int i = 0; i < employeeCount; i++) {  
 employees[i].calcTax();  
 }  
 // Display sales tax for each product  
 System.*out*.println("Sales Tax for Products:");  
 for (int i = 0; i < productCount; i++) {  
 products[i].calcTax();  
 }  
 sc.close(); // close Scanner class object  
 }  
}

Output:

A screenshot of a computer

Description automatically generated