【案例3-1】基于控制台的购书系统

**【案例介绍】**

1. **案例描述**

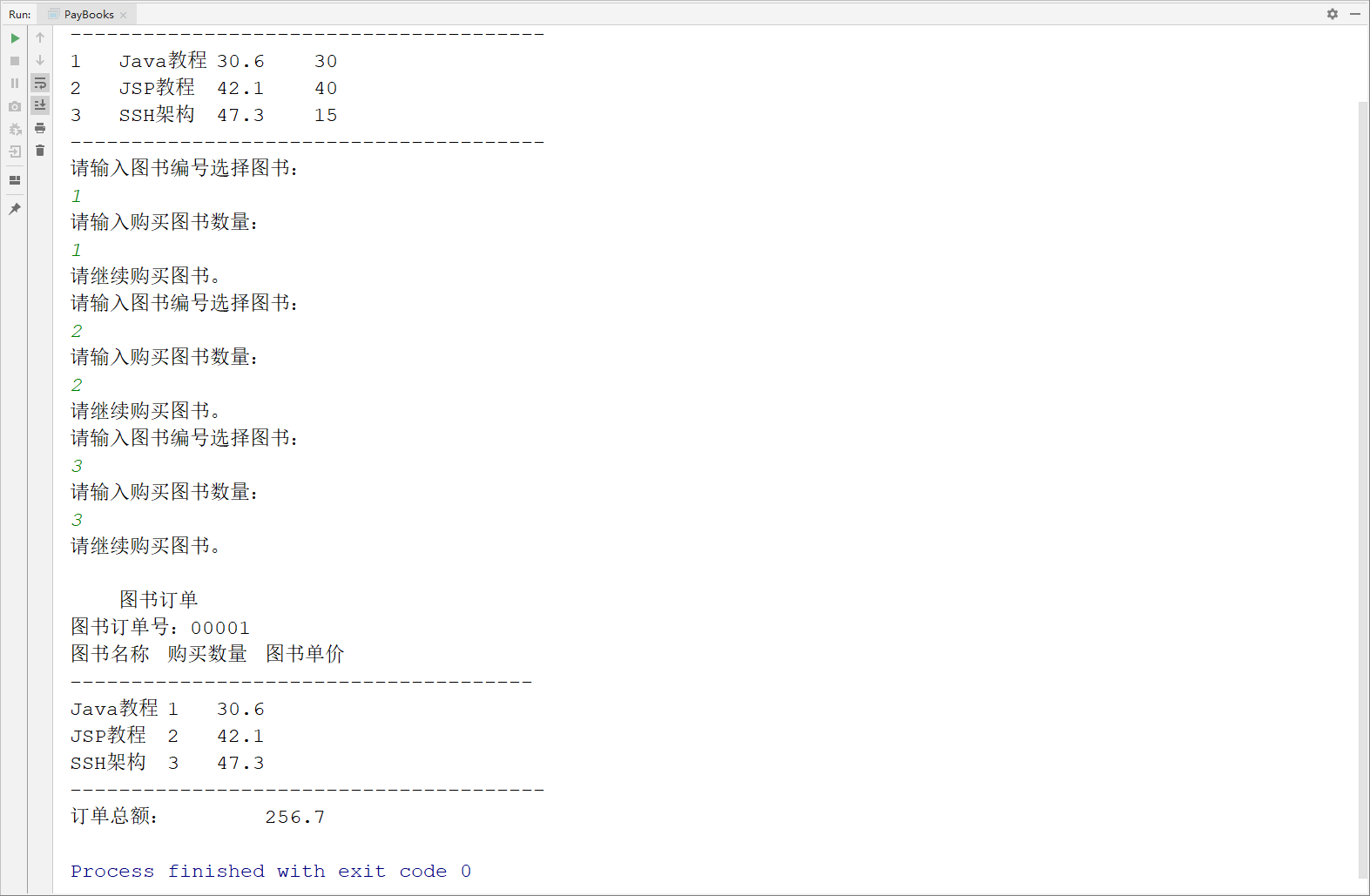
伴随互联网的蓬勃发展，网络购书系统作为电子商务的一种形式，正以其高效、低成本的优势逐步成为新兴的经营模式，人们已经不再满足互联网的用途仅仅局限于信息的浏览和发布，更渴望着能够充分享受互联网所带来的更多便利。网络购书系统正适应了当今社会快节奏地生活，使顾客足不出户便可以方便快捷轻松地选购自己喜欢的图书。

本任务要求，使用所学知识编写一个基于控制台的购书系统，实现购书功能。输出所有图书的信息：包括每本书的编号、书名、单价、库存。

顾客购买书时，根据提示输入图书编号来选购需要的书，并根据提示输入购买书的的数量。

购买完毕后输出顾客的订单信息，包括：订单号、订单明细、订单总额。

1. **运行结果**



**【案例目标】**

* 学会分析“基于控制台的购书系统”程序任务实现的逻辑思路。
* 能独立完成“基于控制台的购书系统”程序的源代码编写、编译及运行。
* 理解和掌握面向对象的设计程序。
* 会用类图进行面向对象设计。
* 掌握封装的实现及好处。
* 包和访问控制修饰符的使用。

**【案例分析】**

（1）通过任务描述可知，该系统中必须包括3个实体类，类名及属性设置如下：

* + - 1. **图书类（Book）：**
         1. 图书编号（id）
         2. 图书名称（name）
         3. 图书单价（price）
         4. 库存数量（storage）
      2. **订单项类（OrderItem）：**
         1. 图书（book）
         2. 购买数量（num）
      3. **订单类（Order）：**
         1. 订单号（orderID）
         2. 订单总额（total）
         3. 订单项列表（items）

（2）由于购买图书时，需要选择图书的数量，所以需要在订单项类里定义获取图书对象以及图书数量的方法。

（3）由于需要指定订单项以及获取订单的订单列表、订单号、订单总额等信息，所以需要有订单列表、订单号、订单总额指定订单项等方法。

**【案例实现】**

定义图书类Book，其代码如下所示。

Book.java

1. public class Book {
2. private int id; // 编号
3. private String name; // 书名
4. private double price; // 价格
5. private int storage; // 库存
6. // 有参构造
7. public Book(int id, String name, double price, int storage) {
8. this.id = id;
9. this.name = name;
10. this.price = price;
11. this.storage = storage;
12. }
13. // 获取书号
14. public int getId() {
15. return id;
16. }
17. // 获取书名
18. public String getName() {
19. return name;
20. }
21. // 获取价格
22. public double getPrice() {
23. return price;
24. }
25. // 获取库存
26. public int getStorage() {
27. return storage;
28. }
29. }

上述代码中，声明了图书标号id、书名name、价格price、库存storage,并提供了其getter和setter方法。

定义订单项类OrderItem，其代码如下所示。

OrderItem.java

1. public class OrderItem {
2. private Book book;
3. private int num;
4. // 有参构造方法
5. public OrderItem(Book book, int num) {
6. this.book = book;
7. this.num = num;
8. }
9. // 获取图书对象
10. public Book getBook() {
11. return book;
12. }
13. // 获取订购图书数量
14. public int getNum() {
15. return num;
16. }
17. }

上述代码中，声明了图书对象Book,购买数量num，并提供了其getter和setter方法。

定义订单类Order，其代码如下所示。

Order.java

1. public class Order {
2. private String orderId;
3. private OrderItem items[];
4. private double total;
5. // 有参构造
6. public Order(String orderId) {
7. this.orderId = orderId;
8. this.items = new OrderItem[3];
9. }
10. // 获取订单号
11. public String getOrderId() {
12. return orderId;
13. }
14. // 获取订单列表
15. public OrderItem[] getItems() {
16. return items;
17. }
18. // 获取订单总额
19. public double getTotal() {
20. calTotal();
21. return total;
22. }
23. // 指定一个订单项
24. public void setItem(OrderItem item, int i) {
25. this.items[i] = item;
26. }
27. // 计算订单总额
28. public void calTotal() {
29. double total = 0;
30. for (int i = 0; i < items.length; i ++) {
31. total += items[i].getNum() \* items[i].getBook().getPrice();
32. }
33. this.total = total;
34. }
35. }

上述代码中，声明了订单id、订单项数组，并声明了获取订单号方法getOrderId()、获取订单列表方法getItems()、获取订单总额方法getTotal()、指定一个订单项setItem()、计算订单总额的方法calTotal()。

PayBooks.java

1. import java.util.Scanner;
2. public class PayBooks {
3. public static void main(String[] args) {
4. Book books[] = new Book[3];
5. //模拟从数据库中读取图书信息并输出
6. outBooks(books);
7. // 顾客购买图书
8. Order order = purchase(books);
9. // 输出订单信息
10. outOrder(order);
11. }
12. // 顾客购买图书
13. public static Order purchase(Book books[]) {
14. Order order = new Order("00001");
15. OrderItem item = null;
16. Scanner in = new Scanner(System.in);
17. for (int i = 0; i < 3; i ++) {
18. System.out.println("请输入图书编号选择图书：");
19. int cno = in.nextInt();
20. System.out.println("请输入购买图书数量：");
21. int pnum = in.nextInt();
22. item = new OrderItem(books[cno-1],pnum);
23. order.setItem(item, i);
24. System.out.println("请继续购买图书。");
25. }
26. in.close();
27. return order;
28. }
29. // 输出订单信息
30. public static void outOrder(Order order) {
31. System.out.println("\n\t图书订单");
32. System.out.println("图书订单号："+order.getOrderId());
33. System.out.println("图书名称\t购买数量\t图书单价");
34. System.out.println("--------------------------------------");
35. OrderItem items[] = order.getItems();
36. for(int i = 0; i < items.length; i ++) {
37. System.out.println(items[i].getBook().getName()+"\t"+items[i].getNum(
38. )+"\t"+items[i].getBook().getPrice());
39. //System.out.println("\n");
40. }
41. System.out.println("---------------------------------------");
42. System.out.println("订单总额：\t\t"+order.getTotal());
43. }
44. // 模拟从数据库中读取图书信息并输出
45. public static void outBooks(Book books[]) {
46. books[0] = new Book(1,"Java教程",30.6,30);
47. books[1] = new Book(2,"JSP教程",42.1,40);
48. books[2] = new Book(3,"SSH架构",47.3,15);
49. System.out.println("\t图书列表");
50. System.out.println("图书编号\t图书名称\t\t图书单价\t库存数量");
51. System.out.println("---------------------------------------");
52. for (int i = 0; i < books.length; i ++) {
53. System.out.println(i+1+"\t"+books[i].getName()+"\t"+
54. books[i].getPrice()+"\t"+books[i].getStorage());
55. }
56. System.out.println("---------------------------------------");
57. }
58. }

上述代码中，输出了图书列表信息包括：图书编号、图书名称、图书单价、库存数量，用户根据图书列表信息，输入图书编号、购买数量等，系统根据用户输入的图书编号及购买数量计算总金额。