比特科技图书管理代码练习

本节目标

> - 熟悉类与类之间的关系
> - 掌握抽象类，接口，继承，封装等知识点的应用

一：简介

本节主要是，利用前面所学的知识点：类，抽象类，封装，继承，多态，接口等进行的一个简单的代码练习。

二：核心需求

1、简单的登录

2、管理端

- 整理书籍(该功能为可扩展功能)
- 查阅书籍
- 增加书籍
- 删除书籍
- 打印书籍列表
- 退出

3、用户端

- 查询书籍
- 借阅书籍
- 归还书籍
- 退出

三: 类的设计

1. 创建图书相关的类

先创建 package book

创建 Book 类, 表示一本书

```
public class Book {
    private String name;
    private String author;
    private int price;
    private String type;
    private boolean isBorrowed = false;


    public Book(String name, String author, int price, String type) {
```

```
        this.name = name;
        this.author = author;
        this.price = price;
        this.type = type;
    }
}
```

创建 BookList 类, 用来保存 N 本书.

```
public class BookList {
    private Book[] books = new Book[10];
    private int size = 0;

    // 弄几个初始值，方便后续测试.
    public BookList() {
        books[0] = new Book("三国演义", "罗贯中", 100,
                "小说");
        books[1] = new Book("水浒传", "施耐庵", 100,
                "小说");
        books[2] = new Book("西游记", "吴承恩", 100,
                "小说");
    }

    public Book getBook(int pos) {
        return books[pos];
    }

    public void setBook(int pos, Book book) {
        books[pos] = book;
    }

    public int getSize() {
        return size;
    }

    public void setSize(int size) {
        this.size = size;
    }
}
```


2. 创建操作相关的类

先创建 package operation

```
public interface IOperation {
    void work(BookList booklist);
}
```

接下来创建一组操作类, 每个类对应一个用户的动作.

```
AddOperation
DelOperation
FindOperation
RemoveOperation
DisplayOperation
BorrowOperation
ReturnOperation
ExitOperation
```

先把空类创建好, 不着急实现细节.

抽象出 Operation 的好处: 让操作和操作之间低耦合, 让操作和用户之间低耦合.

3. 创建用户相关的类

先创建 package user

创建 User 类, 这是一个抽象类

```
// User 类是一个抽象类，每个子类需要做两件事情
// 1. 初始化对应的操作数组
// 2. 实现 Menu 菜单
abstract public class User {
    protected String name;
    protected IOperation[] operations;

    // 显示菜单
    abstract public int menu();
    // 根据用户选项执行操作
    public void doOperation(int choice, BookList bookList) {
        operations[choice].work(bookList);
    }
}
```

创建普通用户类, 是 User 的子类.

```
public class NormalUser extends User {
    public NormalUser(String name) {
        this.name = name;
        this.operations = new IOperation[] {
                new ExitOperation(),
                new FindOperation(),
                new BorrowOperation(),
                new ReturnOperation()
        };
    }

    @Override

    public int menu() {
```

```
        System.out.println("============================");
        System.out.println("Hello " + this.name + ", 欢迎使用图书管理系统!");
        System.out.println("1. 查找图书");
        System.out.println("2. 借阅图书");
        System.out.println("3. 归还图书");
        System.out.println("0. 退出系统");
        System.out.println("============================");
        System.out.println("请输入您的选择: ");
        Scanner scanner = new Scanner(System.in);
        int choice = scanner.nextInt();
        return choice;
    }
}
```

创建管理员用户类

```
public class Admin extends User {
    public Admin(String name) {
        this.name = name;
        this.operations = new IOperation[] {
                new ExitOperation(),
                new FindOperation(),
                new AddOperation(),
                new DelOperation(),
                new DisplayOperation()
        };
    }

    @Override
    public int menu() {
        System.out.println("============================");
        System.out.println("Hello " + this.name + ", 欢迎使用图书管理系统!");
        System.out.println("1. 查找图书");
        System.out.println("2. 新增图书");
        System.out.println("3. 删除图书");
        System.out.println("4. 显示所有图书");
        System.out.println("0. 退出系统");
        System.out.println("============================");
        System.out.println("请输入您的选择: ");
        Scanner scanner = new Scanner(System.in);
        int choice = scanner.nextInt();
        scanner.close();
        return choice;
    }
}
```


4. 进行整合

创建 Main 类和 main 方法, 搭建整体逻辑

```
public class Main {
    public static void main(String[] args) {
        // 1. 准备基本的数据
        BookList bookList = new BookList();
        // 2. 创建用户
        User user = login();
        // 3. 进入主循环
        while (true) {
            int choice = user.menu();
            user.doOperation(choice, bookList);
        }
    }

    public static User login() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("请输入您的姓名:");
        String name = scanner.next();
        System.out.println("请输入您的身份(1 表示管理员, 0 表示普通用户):");
        int who = scanner.nextInt();
        if (who == 1) {
            return new Admin(name);
        }
        return new NormalUser(name);
    }
}
```

可以先测试下代码的基本框架是否存在问题.

5. 实现具体的每个 Operation