ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

****

BÁO CÁO

Bài tập thực hành lab 02

Học phần: Lập trình hướng đối tượng (TN)

Giảng viên hướng dẫn: Lê Thị Hoa

Sinh viên thực hiện: Lê Thị Nhung - 20210662

Mã lớp: 732872

Hà Nội, tháng 10 năm 2023

Mục Lục

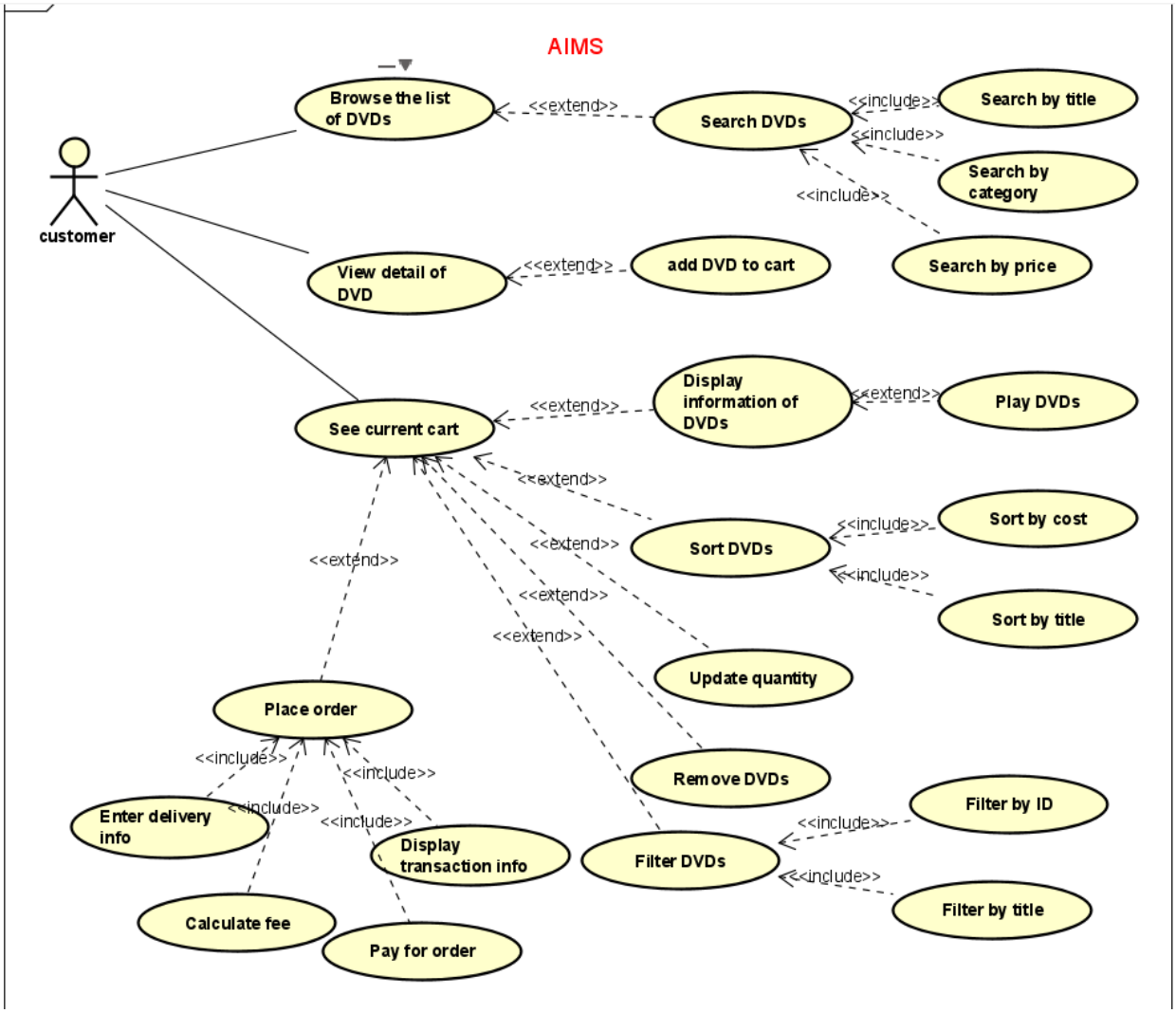
1.Đề bài

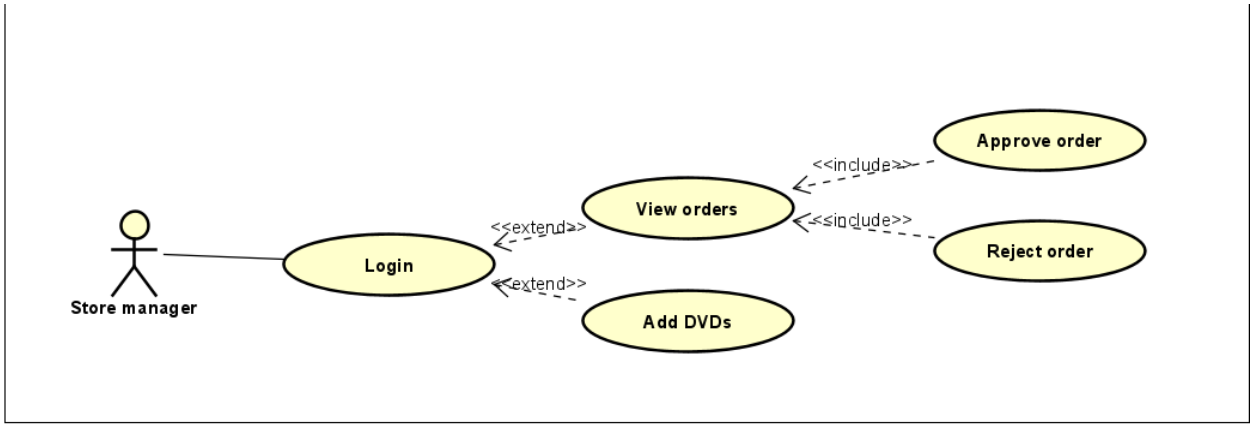
Thiết kế hệ thống mới cho dự án AIMS (HIện tại chỉ có 1 loại phương tiện: DVD)

2.Yêu cầu bài toán

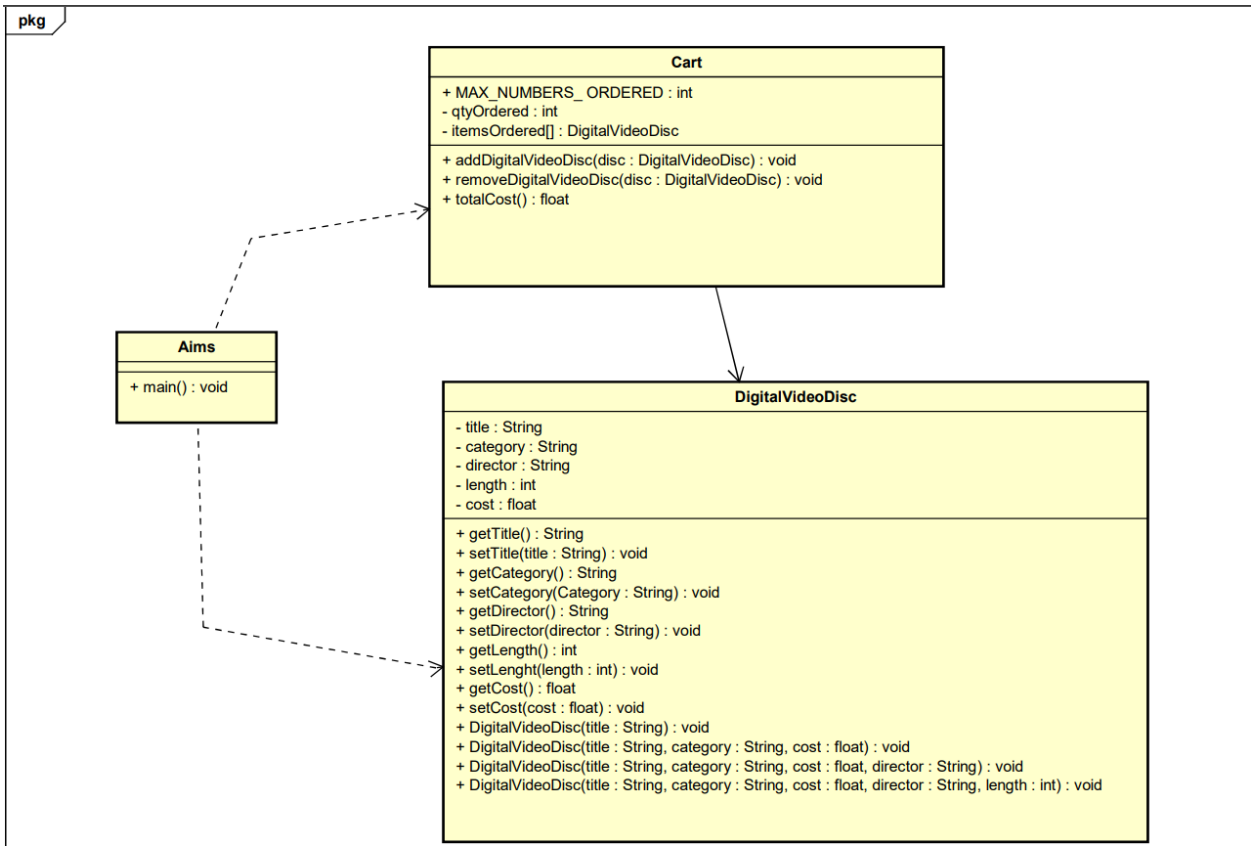
Hệ thống cần tạo một giỏ hàng mới cho người dùng, nơi nó sẽ lưu trữ thông tin về các đĩa DVD mà người dùng muốn mua. Người dùng có thể thêm, xóa đĩa DVD khỏi giỏ hàng cũng như tính toán tổng chi phí. Người dùng có thể thêm tối đa 20 đĩa DVD vào một giỏ hàng. Giỏ hàng cùng với thông tin và hành vi của nó được mô hình hóa bằng lớp Cart. Khi người dùng thêm một đĩa DVD vào giỏ hàng, hệ thống cũng phải tạo ra một đĩa DVD mới dựa trên thông tin mà người dùng cung cấp. Thông tin này có thể được hiển thị bất kỳ lúc nào khi người dùng muốn xem nó. Đĩa DVD cùng với thông tin và chức năng của nó được mô hình hóa bằng lớp DigitalVideoDisc. Cuối cùng, ứng dụng cần một điểm truy cập để hiển thị và nhận đầu vào từ người dùng (thông qua giao diện dòng lệnh)là lớp Aims.

3.Use Case Diagram





4.Class Diagram



5.Mã Nguồn chương trình

5.1 DigitalVideoDisc Class

```java
package lab_02;

public class DigitalVideoDisc { // lưu trữ tiêu đề, danh mục, chi phí, đạo diễn và độ dài
    private String title;
    private String category;
    private String directory;
    private int lenght;
    private float cost;
    public DigitalVideoDisc(String title) {
        super();
        this.title = title;
    }
    public DigitalVideoDisc( String category,String title, float cost) {
        super();
        this.title = title;
        this.category = category;
        this.cost = cost;
    }

    public DigitalVideoDisc(String title, String category, String directory, float cost) {
        super();
        this.title = title;
        this.category = category;
        this.directory = directory;
        this.cost = cost;
    }

    public DigitalVideoDisc(String title, String category, String directory, int lenght, float cost) {
        super();
        this.title = title;
        this.category = category;
        this.directory = directory;
        this.lenght = lenght;
        this.cost = cost;
    }

    public String getTitle() {
        return title;
    }
    public String getCategory() {
        return category;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public void setCategory(String category) {
        this.category = category;
    }
    public void setDirectory(String directory) {
        this.directory = directory;
    }
    public void setLenght(int lenght) {
        this.lenght = lenght;
    }
    public void setCost(float cost) {
        this.cost = cost;
    }
    public String getDirectory() {
        return directory;
    }
    public int getLenght() {
        return lenght;
    }
    public float getCost() {
        return cost;
    }
}
```

5.2 Cart Class

```java
package lab_02;

import java.util.ArrayList;

public class Cart {
    public static final int MAX_NUMBERS_ORDERED = 20;  // số lượng tối đa của giỏ hàng ( hằng số )
    private float totalcost = 0;  // tổng giá trị giỏ hàng
    ArrayList<DigitalVideoDisc> itemsOderedList = new ArrayList<DigitalVideoDisc>(MAX_NUMBERS_ORDERED); // Arraylist lưu danh sách các đĩa DVD
                                                                                                        //trong giỏ hàng
    public int qtyOrdered = 0;  // Số lượng thực tế DVD trong giỏ hàng

    // Thêm 1 dvd vào giỏ hàng
        public void addDigitalVideoDisc(DigitalVideoDisc disc) {
            // Nếu số lượng chưa max thì có thể thêm
            if(qtyOrdered < MAX_NUMBERS_ORDERED) {
                itemsOderedList.add(disc);
                System.out.println("The disc " + disc.getTitle() + " has been added"); // thoong báo rằng đĩa đc lấy tên( disc.getTitle() )
                                                                                        //đã được thêm vào danh sách
                qtyOrdered = itemsOderedList.size();
            }
            // in ra thêm báo số lượng đĩa thêm vào đã đạt giới hạn
            else System.out.println("The cart is almost full");
        }

    // Xóa 1 dvd khỏi giỏ hàng
    public void removeDigitalVideoDisc(DigitalVideoDisc disc) {
        itemsOderedList.remove(disc);
        System.out.println("The disc " + disc.getTitle() + " has been removed");
        qtyOrdered = itemsOderedList.size();
    }
    //Tính tổng giá tiền dvd trong giỏ
    public float totalCost() {
        for(int i = 0; i < itemsOderedList.size(); i++) {
            totalcost += itemsOderedList.get(i).getCost();
        }
        return totalcost;
    }
}
```

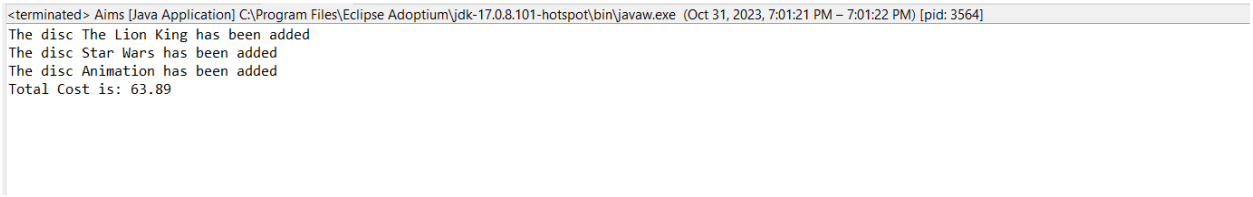
5.3 Aims Class

```java
package lab_02;

public class Aims {

    public static void main(String[] args) {
        // Tạo một giỏi hàng mới
        Cart anOrder = new Cart();

        // Tạo 3 đĩa mới với thông tin: tiêu đề, danh mục, chi phí, đạo diễn và độ dài
        DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King",
                "Animation", "Roger Allers", 87, 19.95f);
        anOrder.addDigitalVideoDisc(dvd1);

        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars",
                "Science Fiction", "George Lucas", 87, 24.95f);
        anOrder.addDigitalVideoDisc(dvd2);

        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin",
                "Animation", 18.99f);

        // add 3 đĩa trên vào giỏi hàng
        anOrder.addDigitalVideoDisc(dvd3);

        //Xoa đĩa thứ 2 ra khỏi giỏ hàng.
        anOrder.removeDigitalVideoDisc(dvd2);


        // tổng giá tiền trả cho giỏ hàng
        System.out.print("Total Cost is: ");
        System.out.println(anOrder.totalCost());
    }

}
```

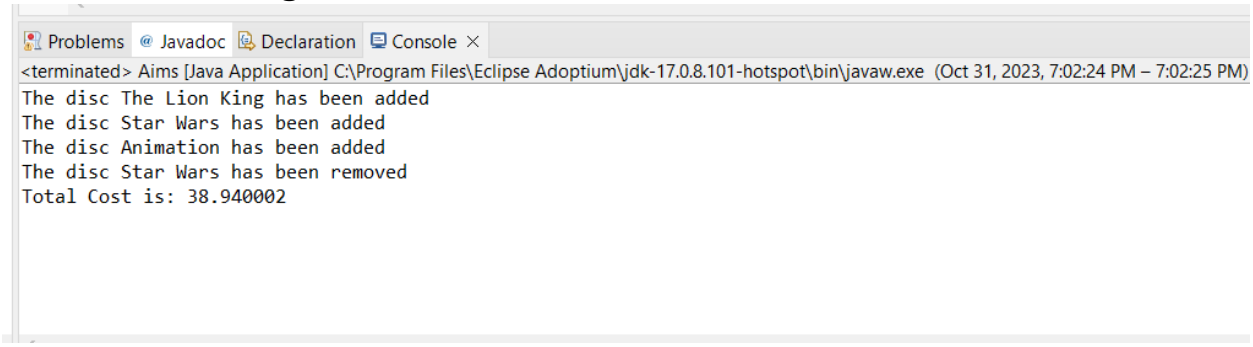
5.4 Chạy chương trình:

5.4.1 Tạo giỏ hàng với số lượng đĩa đã khai báo.

Kết quả chạy được



5.4.2 Xóa đĩa mong muốn



6. Trả lời câu hỏi

If you create a constructor method to build a DVD by title then create a constructor method to build a DVD by category. Does JAVA allow you to do this?

Java allows you to create multiple constructors in a class as long as they have different parameter lists. This is known as constructor overloading, and it is a common practice in Java to provide different ways to initialize objects with different sets of parameters. So, yes, Java allows you to create constructor methods for DVDs based on title and category, provided they have distinct parameter lists, such as different types or a different number of parameters.

When should accessor methods be used?

Encapsulation: Accessor methods are an essential part of the principle of encapsulation in object-oriented programming. They allow you to hide the internal details of an object's state and provide controlled, read-only access to its attributes.

This helps in maintaining data integrity and prevents unintended modification of an object's state.

Data Abstraction: Accessor methods abstract the underlying data and provide a high-level, consistent interface for accessing an object's properties. This makes it easier to work with objects without needing to know the internal implementation details.

Data Validation: Accessor methods can perform validation checks before returning the attribute's value. For example, you can ensure that a temperature attribute is within a valid range or that a string attribute is not null or empty before returning its value.

Computed Properties: Accessor methods can be used to compute and return derived or calculated properties based on the object's existing attributes. For example, you can have a getArea() method in a Rectangle class that computes and returns the area based on its width and height attributes.

Access Control: Accessor methods allow you to control access to an attribute. You can make an attribute private and provide a getter method to allow read-only access while preventing direct modification of the attribute.

Encapsulation of Complex Logic: In some cases, accessor methods may encapsulate complex logic or operations necessary to retrieve the attribute's value. This can include fetching data from a database, performing transformations, or other operations.