Họ và tên: Tạ Tùng Dương MSSV: 20215267

# BÁO CÁO THỰC HÀNH LAB04
# LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

## Mục lục nội dung



## Mục lục hình ảnh



## 1. Creating the abstract Media class

```
1. package LAB04.AimsProject.Media;
```

```
 2.
 3. import java.util.Comparator;
 4.
 5. class MediaComparatorByTitle implements Comparator<Media> {
 6.     @Override
 7.     public int compare(Media media1, Media media2) {
 8.         return media1.getTitle().compareTo(media2.getTitle());
 9.     }
10. }
11.
12. class MediaComparatorByCost implements Comparator<Media> {
13.     @Override
14.     public int compare(Media media1, Media media2) {
15.         return Double.compare(media1.getCost(), media2.getCost());
16.     }
17. }
18.
19. public abstract class Media{
20.     // Attribute
21.     private int id;
22.     private String title;
23.     private String category;
24.     private float cost;
25.     public static final Comparator<Media> COMPARE_BY_TITLE =
26.             new MediaComparatorByTitle();
27.     public static final Comparator<Media> COMPARE_BY_COST    =
28.             new MediaComparatorByCost();
29.
30.     // Constructor
31.     public Media(int id, String title) {
32.         this.id = id;
33.         this.title = title;
34.     }
35.     public Media(int id, String title, String category, float cost) {
36.         this.id = id;
37.         this.title = title;
38.         this.category = category;
39.         this.cost = cost;
40.     }
41.
42.     // Method to print a media
43.     public void print() {
44.     }
45.
46.     // Method to finds out if the corresponding disk is a match given the title.
47.     public boolean isMatch(String title) {
48.         return title.equals(this.title);
49.     }
50.
51.     @Override
52.     public boolean equals(Object o) {
53.         if (o instanceof Media media) {
54.             return title.equals(media.getTitle());
55.         }
56.         return false;
57.     }
58.
59.     // Getter and Setter
60.     public int getId() {
61.         return id;
62.     }
63.
64.     public void setId(int id) {
65.         this.id = id;
66.     }
67.
68.     public String getTitle() {
69.         return title;
70.     }
71.
```

```
72.     public void setTitle(String title) {
73.         this.title = title;
74.     }
75.
76.     public String getCategory() {
77.         return category;
78.     }
79.
80.     public void setCategory(String category) {
81.         this.category = category;
82.     }
83.
84.     public float getCost() {
85.         return cost;
86.     }
87.
88.     public void setCost(float cost) {
89.         this.cost = cost;
90.     }
91. }
92.
```

## 2. Create the Disc class extending the Media class

```
 1. package LAB04.AimsProject.Media;
 2.
 3. public class Disc extends Media{
 4.     // Attribute
 5.     private String director;
 6.     private int length;
 7.
 8.     // Constructor
 9.     public Disc(int id, String title) {
10.         super(id, title);
11.     }
12.
13.     public Disc(int id, String title, String category, float cost, String director, int
length) {
14.         super(id, title, category, cost);
15.         this.director = director;
16.         this.length = length;
17.     }
18.
19.     // Getter and Setter
20.     public String getDirector() {
21.         return director;
22.     }
23.
24.     public void setDirector(String director) {
25.         this.director = director;
26.     }
27.
28.     public int getLength() {
29.         return length;
30.     }
31.
32.     public void setLength(int length) {
33.         this.length = length;
34.     }
35. }
36.
```

## 3. Creating the Playable interface

```
1. package LAB04.AimsProject.Media;
2.
3. public interface Playable {
4.     // Method to play
```

```
    public void play();
}
```

## 4. Creating the Book class

```
package LAB04.AimsProject.Media;

import java.util.ArrayList;
import java.util.List;

public class Book extends Media{
    // Attribute
    private List<String> authors = new ArrayList<String>();

    // Constructor
    public Book(int id, String title, String category, float cost) {
        super(id, title, category, cost);
    }

    // Method to add an author
    public void addAuthor(String authorName) {
        int indexOfAuthor = authors.indexOf(authorName);
        if (indexOfAuthor == -1) {
            System.out.println("Author is already in the list");
            return;
        }
        authors.add(authorName);
        System.out.println("Added");
    }

    // Method to remove an author
    public void removeAuthor(String authorName) {
        int indexOfAuthor = authors.indexOf(authorName);
        if (indexOfAuthor == -1) {
            System.out.println("Author is absent in the list");
            return;
        }
        authors.remove(indexOfAuthor);
        System.out.println("Removed");
    }

    // Method to print a book
    @Override
    public void print() {
        System.out.print(getId() + ". Book - "
                + getTitle() + " - "
                + getCategory() + " - ");
        for (String author: authors) {
            System.out.print(author + " - ");
        }
        System.out.println(getCost() + "$");
    }

    // Getter and Setter
    public List<String> getAuthors() {
        return authors;
    }

    public void setAuthors(List<String> authors) {
        this.authors = authors;
    }
}
```

## 5. Creating the Track class

```
package LAB04.AimsProject.Media;
```

```java

public class Track implements Playable{
    // Attribute
    private String title;
    private int length;

    // Constructor
    public Track(String title, int length) {
        this.title = title;
        this.length = length;
    }

    // Method to play a track
    public void play() {
        System.out.println("Playing track: " + title);
        System.out.println("Track length : " + length);
    }

    @Override
    public boolean equals(Object o) {
        if (o instanceof Track track) {
            return title.equals(track.getTitle()) && length == track.getLength();
        }
        return false;
    }

    // Getter and Setter
    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public int getLength() {
        return length;
    }

    public void setLength(int length) {
        this.length = length;
    }
}

```

## 6. Creating the CompactDisc class

```java
package LAB04.AimsProject.Media;

import java.util.ArrayList;
import java.util.List;

public class CompactDisc extends Disc implements Playable{
    // Attribute
    private String artist;
    private List<Track> tracks = new ArrayList<Track>();

    // Constructor
    public CompactDisc(int id, String title, String category, float cost,
                       String director, int length, String artist) {
        super(id, title, category, cost, director, length);
        this.artist = artist;
    }

    // Method to add a track
    public void addTrack(Track track) {
        int indexOfTrack = tracks.indexOf(track);
        if (indexOfTrack == -1) {
            System.out.println("Track is already in the list");
```

```
            return;
        }
        tracks.add(track);
        System.out.println("Added");
    }

    // Method to remove a track
    public void removeTrack(Track track) {
        int indexOfTrack = tracks.indexOf(track);
        if (indexOfTrack == -1) {
            System.out.println("Track is absent in the list");
            return;
        }
        tracks.remove(indexOfTrack);
        System.out.println("Removed");
    }

    // Method to get the length of CD
    @Override
    public int getLength() {
        int length = 0;
        for (Track track: tracks) {
            length += track.getLength();
        }
        setLength(length);
        return length;
    }

    // Method to play a track
    public void play() {
        System.out.println("Playing CD: " + this.getTitle());
        System.out.println("CD artist: " + artist);
        System.out.println("CD length: " + this.getLength());
        for (Track track: tracks) {
            track.play();
        }
    }

    // Method to print a cd
    @Override
    public void print() {
        System.out.println(getId() + ". CD - "
                + getTitle() + " - "
                + getCategory() + " - "
                + getDirector() + " - "
                + artist + " - "
                + getLength() + ": "
                + getCost() + "$");
    }

    // Getter and Setter
    public String getArtist() {
        return artist;
    }

    public void setArtist(String artist) {
        this.artist = artist;
    }
}

```

## 7. Update the DigitalVideoDisc class

```
package LAB04.AimsProject.Media;

public class DigitalVideoDisc extends Disc implements Playable{
    // Constructor
    public DigitalVideoDisc(int id, String title) {
        super(id, title);
```

```
    }

    public DigitalVideoDisc(int id, String title, String category, float cost) {
        this(id, title);
        this.setCategory(category);
        this.setCost(cost);
    }

    public DigitalVideoDisc(int id, String title, String category, String director, float
cost) {
        this(id, title, category, cost);
        this.setDirector(director);
    }

    public DigitalVideoDisc(int id, String title, String category, String director, int
length, float cost) {
        this(id, title, category, director, cost);
        this.setLength(length);
    }

    // Method to print a dvd
    @Override
    public void print() {
        System.out.println(getId() + ". DVD - "
                + getTitle() + " - "
                + getCategory() + " - "
                + getDirector() + " - "
                + getLength() + ": "
                + getCost() + "$");
    }

    // Method to play a dvd
    public void play() {
        System.out.println("Playing DVD: " + this.getTitle());
          System.out.println("DVD length: " + this.getLength());
    }
}

```

## 8. Update the Cart class

```
package LAB04.AimsProject;

import LAB04.AimsProject.Media.DigitalVideoDisc;
import LAB04.AimsProject.Media.Media;
import java.util.ArrayList;
import java.util.List;

public class Cart {
    // Attribute
    private List<Media> itemsOrdered = new ArrayList<Media>();
    private int numOfDVDs;

    // Constructor
    public Cart() {
        numOfDVDs = 0;
    }

    // Method to add a new media
    public void addMedia(Media media) {
        if (itemsOrdered.contains(media)) {
            System.out.println("Media is already in the list");
            return;
        }
        itemsOrdered.add(media);
        if (media.getClass() == DigitalVideoDisc.class) {
            numOfDVDs++;
        }
        System.out.println("Added");
```

```
29.     }
30.
31.     // Method to remove a media
32.     public void removeMedia(Media media) {
33.         // Search for media
34.         int indexOfRemoved = itemsOrdered.indexOf(media);
35.
36.         // If not found
37.         if (indexOfRemoved == -1) {
38.             System.out.println("Not found");
39.             return;
40.         }
41.
42.         // Remove
43.         itemsOrdered.remove(indexOfRemoved);
44.         if (media.getClass() == DigitalVideoDisc.class) {
45.             numOfDVDs--;
46.         }
47.
48.         // Notify
49.         System.out.println("Removed");
50.     }
51.
52.     // Method to calculate the total cost
53.     public double totalCost() {
54.         float cost = 0;
55.         for (Media media: itemsOrdered) {
56.             cost += media.getCost();
57.         }
58.
59.         return Math.round(cost * 100.0) / 100.0;
60.     }
61.
62.     // Method to print the list of ordered items of a cart,
63.     // the price of each item, and the total price
64.     public void printCart() {
65.         System.out.println("**********************CART**********************");
66.         System.out.println("Ordered Items:");
67.         for (Media media : itemsOrdered) {
68.             media.print();
69.         }
70.         System.out.println("Total cost: " + totalCost());
71.         System.out.println("************************************************");
72.     }
73.
74.     // Method to search for media in the cart by ID and display the search results.
75.     public Media searchByID(int id) {
76.         for (Media media: itemsOrdered) {
77.             if (media.getId() == id) {
78.                 return media;
79.             }
80.         }
81.         System.out.println("Not found!");
82.         return null;
83.     }
84.
85.     // Method to search for media in the cart by title.
86.     public Media searchByTitle(String title) {
87.         for (Media media: itemsOrdered) {
88.             if (media.isMatch(title)) {
89.                 return media;
90.             }
91.         }
92.         System.out.println("Not found!");
93.         return null;
94.     }
95.
96.     // Method to sort by title and print
97.     public void sortByTitle() {
98.         itemsOrdered.sort(Media.COMPARE_BY_TITLE);
```

```
 99.         printCart();
100.     }
101.
102.     // Method to sort by cost and print
103.     public void sortByCost() {
104.         itemsOrdered.sort(Media.COMPARE_BY_COST);
105.         printCart();
106.     }
107.
108.     // Getter and Setter
109.
110.     public int getNumOfDVDs() {
111.         return numOfDVDs;
112.     }
113.
114.     public void setNumOfDVDs(int numOfDVDs) {
115.         this.numOfDVDs = numOfDVDs;
116.     }
117. }
118.
```

## 9. Update the Store class

```
 1. package LAB04.AimsProject;
 2.
 3. import LAB04.AimsProject.Media.Media;
 4.
 5. import java.util.ArrayList;
 6. import java.util.List;
 7.
 8. public class Store {
 9.     // Attribute
10.     private List<Media> itemsInStore = new ArrayList<Media>();
11.
12.     // Constructor
13.     public Store() {
14.     }
15.
16.     // Method to add a media
17.     public void addMedia(Media media) {
18.         if (itemsInStore.contains(media)) {
19.             System.out.println("Media is already in the list");
20.             return;
21.         }
22.         itemsInStore.add(media);
23.         System.out.println("Added");
24.     }
25.
26.     // Method to remove a media
27.     public void removeMedia(Media media) {
28.         // Search for disc
29.         int indexOfRemoved = itemsInStore.indexOf(media);
30.
31.         // If not found
32.         if (indexOfRemoved == -1) {
33.             System.out.println("Not found");
34.             return;
35.         }
36.
37.         // Remove
38.         itemsInStore.remove(indexOfRemoved);
39.
40.         // Notify
41.         System.out.println("Removed");
42.     }
43.
44.     // Method to print all item in store
45.     public void printStore() {
46.         System.out.println("**********************STORE**********************");
```

```
47.         System.out.println("Items in store:");
48.         for (Media media : itemsInStore) {
49.             media.print();
50.         }
51.         System.out.println("**********************************************");
52.     }
53.
54.     // Method to search for media in the store by title.
55.     public Media searchByTitle(String title) {
56.         for (Media media: itemsInStore) {
57.             if (media.isMatch(title)) {
58.                 return media;
59.             }
60.         }
61.         System.out.println("Not found!");
62.         return null;
63.     }
64.
65.     // Getter and Setter
66.     public List<Media> getItemsInStore() {
67.         return itemsInStore;
68.     }
69.
70.     public void setItemsInStore(List<Media> itemsInStore) {
71.         this.itemsInStore = itemsInStore;
72.     }
73. }
74.
```

## 10. Update the Aims class

```
 1. package LAB04.AimsProject;
 2.
 3. import LAB04.AimsProject.Cart;
 4. import LAB04.AimsProject.Media.*;
 5.
 6. import java.util.Scanner;
 7.
 8. public class Aims {
 9.     // Create a new store
10.     static Store store = new Store();
11.     // Create a new cart
12.     static Cart cart = new Cart();
13.     public static void showMenu() {
14.         int command;
15.         do {
16.             Scanner scanner = new Scanner(System.in);
17.             // Show store
18.             System.out.println("AIMS: ");
19.             System.out.println("--------------------------------");
20.             System.out.println("1. View store");
21.             System.out.println("2. Update store");
22.             System.out.println("3. See current cart");
23.             System.out.println("0. Exit");
24.             System.out.println("--------------------------------");
25.             System.out.println("Please choose a number: 0-1-2-3");
26.             command = scanner.nextInt();
27.
28.             // If chose View store
29.             if (command == 1) {
30.                 store.printStore();
31.                 storeMenu();
32.             }
33.
34.             // If chose 2
35.             if (command == 2) {
36.                 updateStoreMenu();
37.             }
38.
```

```
            // If chose 3
            if (command == 3) {
                cart.printCart();
                cartMenu();
            }
        } while (command != 0);
    }

    public static void updateStoreMenu() {
        Scanner scanner = new Scanner(System.in);
        // Show menu
        System.out.println("Options: ");
        System.out.println("--------------------------------");
        System.out.println("1. Add a media to store");
        System.out.println("2. Remove a media from cart");
        System.out.println("0. Back");
        System.out.println("--------------------------------");
        System.out.println("Please choose a number: 0-1-2");
        int command = scanner.nextInt();

        // If chose 1
        if (command == 1) {
            System.out.println("Added to store");
        }

        // If chose 2
        if (command == 2) {
            removeMediaFromStore();
        }
    }

    public static void storeMenu() {
        Scanner scanner = new Scanner(System.in);
        // Show menu
              System.out.println("Options: ");
              System.out.println("--------------------------------");
              System.out.println("1. See a media's details");
              System.out.println("2. Add a media to cart");
        System.out.println("3. Play a media");
              System.out.println("4. See current cart");
              System.out.println("0. Back");
              System.out.println("--------------------------------");
              System.out.println("Please choose a number: 0-1-2-3-4");
        int command = scanner.nextInt();

        // If chose 1
        if (command == 1) {
            Media media;
            do {
                System.out.println("Enter the title of the media: ");
                scanner.nextLine();
                String title = scanner.nextLine();
                media = store.searchByTitle(title);
            } while (media == null);
            media.print();
            mediaDetailsMenu(media);
        }

        // If chose 2
        if (command == 2) {
            addMediaToCart();
        }

        // If chose 3
        if (command == 3) {
            Media media;
            do {
                System.out.println("Enter the title of the media: ");
                scanner.nextLine();
                String title = scanner.nextLine();
```

```
                media = store.searchByTitle(title);
            } while (media == null);
            playAMedia(media);
        }

        // If chose 4
        if (command == 4) {
            cart.printCart();
            cartMenu();
        }
    }

    public static void mediaDetailsMenu(Media media) {
        Scanner scanner = new Scanner(System.in);
        // Show menu
        System.out.println("Options: ");
        System.out.println("--------------------------------");
        System.out.println("1. Add to cart");
        if (!(media instanceof Book)) {
            System.out.println("2. Play");
        }
        System.out.println("0. Back");
        System.out.println("--------------------------------");
        System.out.print("Please choose a number: 0-1");
        if (!(media instanceof Book)) {
            System.out.println("-2");
        }
        int command = scanner.nextInt();

        // If chose 1
        if (command == 1) {
            cart.addMedia(media);
        }

        // If chose 2
        if (command == 2) {
            playAMedia(media);
        }
    }

    public static void cartMenu() {
        Scanner scanner = new Scanner(System.in);
        // Show menu
        System.out.println("Options: ");
              System.out.println("--------------------------------");
              System.out.println("1. Filter medias in cart");
              System.out.println("2. Sort medias in cart");
              System.out.println("3. Remove media from cart");
              System.out.println("4. Play a media");
              System.out.println("5. Place order");
              System.out.println("0. Back");
              System.out.println("--------------------------------");
              System.out.println("Please choose a number: 0-1-2-3-4-5");
        int command = scanner.nextInt();

        // If chose 1
        if (command == 1) {
            filterCartMenu();
        }

        // If chose 2
        if (command == 2) {
            sortCartMenu();
        }

        // If chose 3
        if (command == 3) {
            removeMediaFromCart();
        }

```

```
179.         // If chose 4
180.         if (command == 4) {
181.             Media media;
182.             do {
183.                 System.out.println("Enter the title of the media: ");
184.                 String title = scanner.nextLine();
185.                 media = cart.searchByTitle(title);
186.             } while (media == null);
187.             playAMedia(media);
188.         }
189. 
190.         // If chose 5
191.         if (command == 5) {
192.             System.out.println("Order is created");
193.             cart = new Cart();
194.         }
195.     }
196. 
197.     public static void filterCartMenu() {
198.         Scanner scanner = new Scanner(System.in);
199.         // Show menu
200.         System.out.println("Options: ");
201.         System.out.println("--------------------------------");
202.         System.out.println("1. By id");
203.         System.out.println("2. By title");
204.         System.out.println("0. Back");
205.         System.out.println("--------------------------------");
206.         System.out.println("Please choose a number: 0-1-2");
207.         int command = scanner.nextInt();
208. 
209.         // If chose 1
210.         if (command == 1) {
211.             System.out.println("Enter the id: ");
212.             int id = scanner.nextInt();
213.             Media media = cart.searchByID(id);
214.             if (media != null) {
215.                 media.print();
216.             }
217.         }
218. 
219.         // If chose 2
220.         if (command == 2) {
221.             System.out.println("Enter the title: ");
222.             scanner.nextLine();
223.             String title = scanner.nextLine();
224.             Media media = cart.searchByTitle(title);
225.             if (media != null) {
226.                 media.print();
227.             }
228.         }
229.     }
230. 
231.     public static void sortCartMenu() {
232.         Scanner scanner = new Scanner(System.in);
233.         // Show menu
234.         System.out.println("Options: ");
235.         System.out.println("--------------------------------");
236.         System.out.println("1. By title");
237.         System.out.println("2. By cost");
238.         System.out.println("0. Back");
239.         System.out.println("--------------------------------");
240.         System.out.println("Please choose a number: 0-1-2");
241.         int command = scanner.nextInt();
242. 
243.         // If chose 1
244.         if (command == 1) {
245.             cart.sortByTitle();
246.         }
247. 
248.         // If chose 2
```

```
249.         if (command == 2) {
250.             cart.sortByCost();
251.         }
252.     }
253.
254.     public static void removeMediaFromCart() {
255.         Scanner scanner = new Scanner(System.in);
256.         Media media;
257.         do {
258.             System.out.println("Enter the title of the media: ");
259.             String title = scanner.nextLine();
260.             media = cart.searchByTitle(title);
261.         } while (media == null);
262.         cart.removeMedia(media);
263.     }
264.
265.     public static void removeMediaFromStore() {
266.         Scanner scanner = new Scanner(System.in);
267.         Media media;
268.         do {
269.             System.out.println("Enter the title of the media: ");
270.             String title = scanner.nextLine();
271.             media = store.searchByTitle(title);
272.         } while (media == null);
273.         store.removeMedia(media);
274.     }
275.
276.     public static void playAMedia(Media media) {
277.         if (media instanceof DigitalVideoDisc dvd) {
278.             dvd.play();
279.         } else if (media instanceof CompactDisc cd) {
280.             cd.play();
281.         }
282.     }
283.
284.     public static void addMediaToCart() {
285.         Scanner scanner = new Scanner(System.in);
286.         Media media;
287.         do {
288.             System.out.println("Enter the title of the media: ");
289.             String title = scanner.nextLine();
290.             media = store.searchByTitle(title);
291.         } while (media == null);
292.         cart.addMedia(media);
293.         if (media instanceof DigitalVideoDisc) {
294.             System.out.println("Number of DVDs in the current cart: " +
cart.getNumOfDVDs());
295.         }
296.     }
297.
298.     public static void main(String[] args) {
299.         // Create new media and add them to the store
300.         // Adding DVDs
301.         Media dvd1 = new LAB04.AimsProject.Media.DigitalVideoDisc(1, "Inception",
302.                 "Science Fiction", "Christopher Nolan", 148, 19.99f);
303.         store.addMedia(dvd1);
304.
305.         Media dvd2 = new LAB04.AimsProject.Media.DigitalVideoDisc(2, "The Dark Knight",
306.                 "Action", "Christopher Nolan", 152, 17.99f);
307.         store.addMedia(dvd2);
308.
309.         Media dvd3 = new DigitalVideoDisc(7, "Interstellar",
310.                 "Science Fiction", "Christopher Nolan", 169, 21.99f);
311.         store.addMedia(dvd3);
312.
313.         // Adding CDs
314.         Media cd1 = new CompactDisc(3, "Random Access Memories",
315.                 "Electronic", 15.99f, "Daft Punk", 13, "Daft Punk");
316.         store.addMedia(cd1);
317.
```

```
318.         Media cd2 = new CompactDisc(4, "25",
319.                 "Pop", 14.99f, "Adele", 11, "Adele");
320.         store.addMedia(cd2);
321.
322.         Media cd3 = new CompactDisc(8, "Lover",
323.                 "Pop", 17.99f, "Taylor Swift", 18, "Taylor Swift");
324.         store.addMedia(cd3);
325.
326.         // Adding Books
327.         Media book1 = new Book(5, "The Silent Patient",
328.                 "Thriller", 14.95f);
329.         store.addMedia(book1);
330.
331.         Media book2 = new Book(6, "Where the Crawdads Sing",
332.                 "Mystery", 12.99f);
333.         store.addMedia(book2);
334.
335.         Media book3 = new Book(9, "Educated",
336.                 "Memoir", 16.95f);
337.         store.addMedia(book3);
338.
339.         Media book4 = new Book(10, "Becoming",
340.                 "Autobiography", 22.99f);
341.         store.addMedia(book4);
342.         showMenu();
343.     }
344. }
345.
```

## 11. Demo Aims

```
--- exec:3.1.0:exec (default-cli) @ OOP ---
Added
Added
Added
Added
Added
Added
Added
Added
Added
Added
```

*Figure 1: DemoAims_Add media to Store*

```
AIMS:
--------------------------------
1. View store
2. Update store
3. See current cart
0. Exit
--------------------------------
Please choose a number: 0-1-2-3
```

*Figure 2: DemoAims_Menu Options*

```
AIMS:
--------------------------------
1. View store
2. Update store
3. See current cart
0. Exit
--------------------------------
Please choose a number: 0-1-2-3
1
************************STORE************************
Items in store:
1. DVD - Inception - Science Fiction - Christopher Nolan - 148: 19.99$
2. DVD - The Dark Knight - Action - Christopher Nolan - 152: 17.99$
7. DVD - Interstellar - Science Fiction - Christopher Nolan - 169: 21.99$
3. CD - Random Access Memories - Electronic - Daft Punk - Daft Punk - 0: 15.99$
4. CD - 25 - Pop - Adele - Adele - 0: 14.99$
8. CD - Lover - Pop - Taylor Swift - Taylor Swift - 0: 17.99$
5. Book - The Silent Patient - Thriller - 14.95$
6. Book - Where the Crawdads Sing - Mystery - 12.99$
9. Book - Educated - Memoir - 16.95$
10. Book - Becoming - Autobiography - 22.99$
*****************************************************
```

*Figure 3: DemoAims_View store*

```
**********STORE**********
Items in store:
1. DVD - Inception - Science Fiction - Christopher Nolan - 148: 19.99$
2. DVD - The Dark Knight - Action - Christopher Nolan - 152: 17.99$
7. DVD - Interstellar - Science Fiction - Christopher Nolan - 169: 21.99$
3. CD - Random Access Memories - Electronic - Daft Punk - Daft Punk - 0: 15.99$
4. CD - 25 - Pop - Adele - Adele - 0: 14.99$
8. CD - Lover - Pop - Taylor Swift - Taylor Swift - 0: 17.99$
5. Book - The Silent Patient - Thriller - 14.95$
6. Book - Where the Crawdads Sing - Mystery - 12.99$
9. Book - Educated - Memoir - 16.95$
10. Book - Becoming - Autobiography - 22.99$
**************************************************
Options:
--------------------------------
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
--------------------------------
Please choose a number: 0-1-2-3-4
1
Enter the title of the media:
Inception
1. DVD - Inception - Science Fiction - Christopher Nolan - 148: 19.99$
Options:
--------------------------------
1. Add to cart
2. Play
0. Back
--------------------------------
Please choose a number: 0-1-2
1
Added
```

*Figure 4: DemoAims_View detail of a media and add to cart*

```
AIMS:
--------------------------------
1. View store
2. Update store
3. See current cart
0. Exit
--------------------------------
Please choose a number: 0-1-2-3
0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  06:54 min
Finished at: 2023-12-01T17:58:44+07:00
------------------------------------------------------------------------
```

*Figure 5: DemoAims_Exit program*

## 12. Update the class diagram

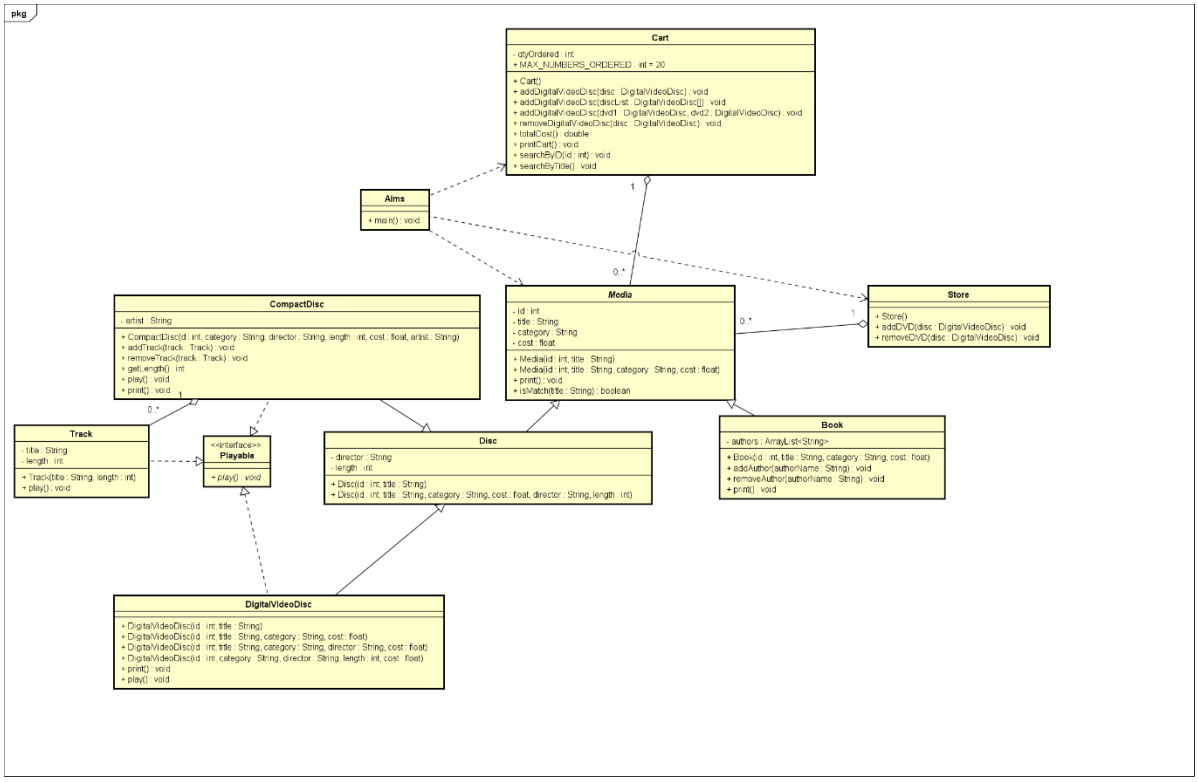


*Figure 6: Update the class diagram*

## 13. Update the usecase diagram

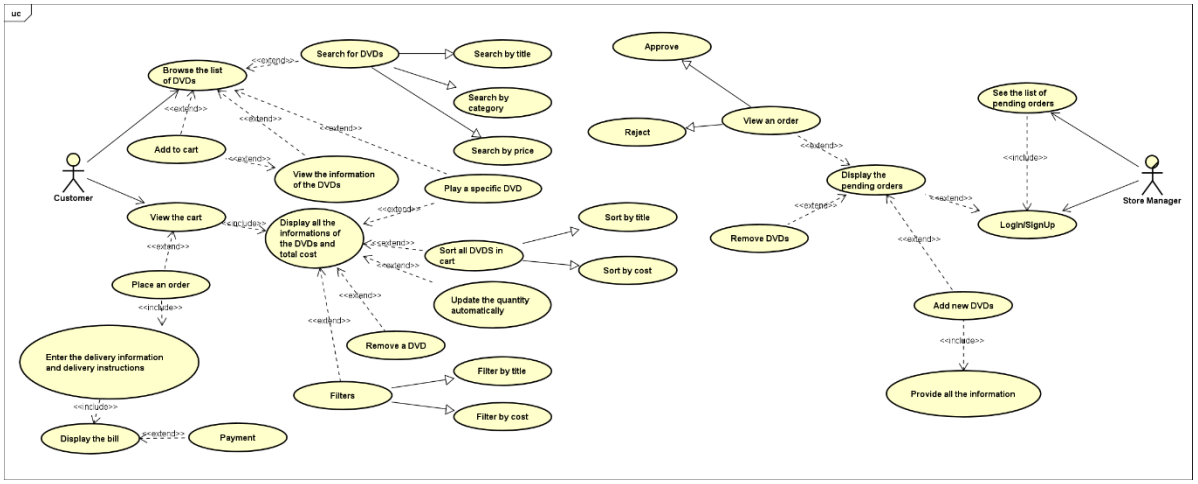


*Figure 7: Update the usecase diagram*

## 14. Answer the questions

- Trong trường hợp cần so sánh các Media với nhau bằng cách implement Comparable thay vì Comparator, thì thay vì tạo class riêng cho từng Comparator, ta cần cho class Media implement interface Comparable.

- Ví dụ như sau:

```
1. public abstract class Media implements Comparable<Media> {
2.     @Override
3.     public int compareTo(Media otherMedia) {
4.         // Compare by title
5.         return this.title.compareTo(otherMedia.getTitle());
6.     }
7. }
```

- Có thể, cài đặt như sau:

```
 1. public abstract class Media implements Comparable<Media> {
 2.     @Override
 3.     public int compareTo(Media otherMedia) {
 4.         // Compare by title first
 5.         int titleComparison = this.title.compareTo(otherMedia.getTitle());
 6.
 7.         // If titles are equal, compare by cost
 8.         return (titleComparison == 0) ? Float.compare(this.cost, otherMedia.getCost()) :
titleComparison;
 9.     }
10. }
```

- Cài đặt như sau:

```
 1. public class DVD extends Media {
 2.     // Override compareTo for DVDs
 3.     @Override
 4.     public int compareTo(Media otherMedia) {
 5.         if (otherMedia instanceof DVD) {
 6.             DVD otherDVD = (DVD) otherMedia;
 7.
 8.             // Compare by title first
 9.             int titleComparison = getTitle().compareTo(otherDVD.getTitle());
10.
11.             // If titles are equal, compare by decreasing length
12.             if (titleComparison == 0) {
13.                 int lengthComparison = Integer.compare(otherDVD.getLength(), getLength());
14.
15.                 // If lengths are equal, compare by cost
16.                 return (lengthComparison == 0) ? Float.compare(getCost(),
otherDVD.getCost()) : lengthComparison;
17.             }
18.
19.             return titleComparison;
20.         } else {
21.             // For non-DVD media, use the default comparison (title then cost)
22.             return super.compareTo(otherMedia);
23.         }
24.     }
25. }
26.
```