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

BÁO CÁO THỰC HÀNH
744474_IT3103_2024.1

BÀI THỰC HÀNH 04

Họ và tên : Đặng Công Đức
MSSV: 20225809
Lớp: **K67 – Việt Nhật 01**

Hà Nội 11/2024

BÁO CÁO THỰC HÀNH LAB 4 LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Contents

Table of Figures

1 Create the Book class

```java
import java.util.*;

public class Book {
    private String title;
    private String category;
    private int id;
    private float cost;
    private List<String> authors = new ArrayList<String>();
    private Book(int id, String title, String category, float cost){
}

public Book(int id, String title, String category, float cost, List<String> authors){

}

public List<String> getAuthors() {
        return authors;
    }

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

    public void addAuthor(String authorName) {
        if (!authors.contains(authorName)) {
            authors.add(authorName);
            System.out.println(authorName+" has been added to the list.");
        }
        System.out.println(authorName+" is already in the list.");
    }

```

Figure 1.1: Book Class 1

```
34      public void removeAuthor(String authorName) {
35          if (!authors.contains(authorName)) {
36              System.out.println(authorName+" is not in the list.");
37          }
38          else {
39              authors.remove(authorName);
40              System.out.println(authorName+" has been removed from the list.");
41          }
42      }
43  }
```

Figure 1.2: Book Class 2

2 Creating the abstract Media class

Đây sẽ là lớp cha để các lớp DigitalVideoDisc, Book kế thừa.

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;

public abstract class Media {
    private int id;
    private String title;
    private String category;
    private float cost;

    public Media(int id, String title, String category, float cost) {
        this.id = id;
        this.title = title;
        this.category = category;
        this.cost = cost;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }

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

Figure 2.1: Media Class 1

```java
    public void setId(int id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }

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

    public String getCategory() {
        return category;
    }

    public void setCategory(String category) {
        this.category = category;
    }

    public float getCost() {
        return cost;
    }

    public void setCost(float cost) {
        this.cost = cost;
    }
}
```

Figure 2.2: Media Class 2

3 Creating the CompactDisc class

3.1 Create the Disc class extending the Media clas

```java

public class Disc extends Media {
    private int length;
    private String director;

    public Disc(int id, String title, String category, float cost) {
        super(id,title, category, cost);
    }

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

    public int getLength() {
        return length;
    }
    public void setLength(int length) {
        this.length = length;
    }
    public String getDirector() {
        return director;
    }
    public void setDirector(String director) {
        this.director = director;
    }

}
```

Figure 3.1: Disc Class

Lab04 > AimsProject > DigitalVideoDisc.java

```java
public class DigitalVideoDisc extends Disc implements Playable {
    public DigitalVideoDisc(int id, String title, String category, float cost) {
        super(id, title, category, cost);
    }
```

Figure 3.2: DigitalVideoDisc Class

```java
package hust.soict.itep.aims.media;

// CompactDisc class represents a CD and extends the Disc class,
15 usages  Vietanh-Vu *  7 related problems
public class CompactDisc extends Disc  {

}
```

Figure 3.3: CompactDisc Class

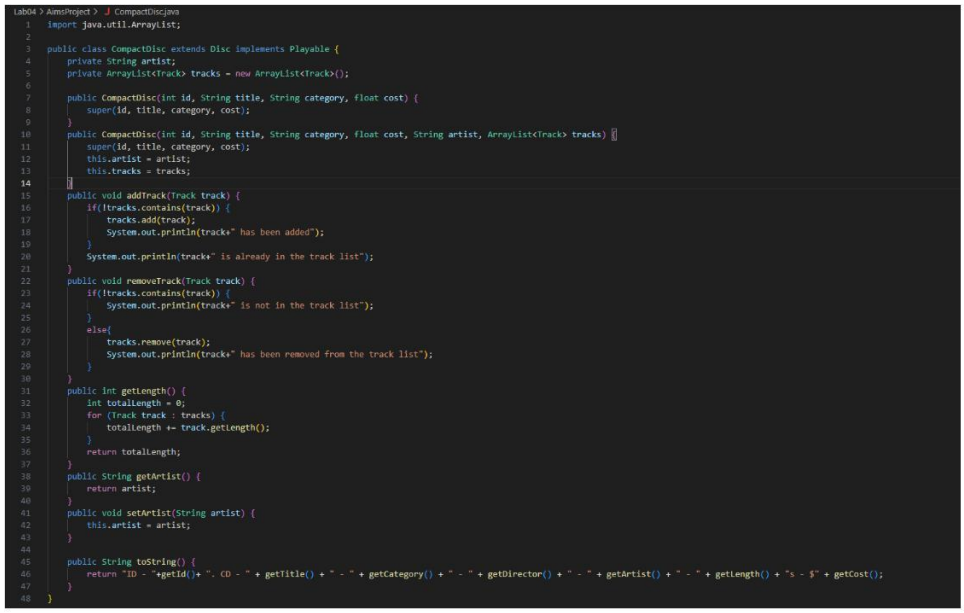
3.2 Create the Track class which models a track on a compact disc and will store information incuding the title and length of the track.

Lab04 > AimsProject > Track.java

```java
public class Track extends CompactDisc implements Playable {
    private String titleTrack;
    private int length;

    public Track(int id, String title, String category, float cost, String titleTrack, int length) {
        super(id, title, category, cost);
        this.titleTrack = titleTrack;
        this.length = length;
    }

    public String getTitle() {
        return titleTrack;
    }
    public void setTitle(String title) {
        this.titleTrack = title;
    }
    public int getLength() {
        return length;
    }
    public void setLength(int length) {
        this.length = length;
    }

    public boolean equals(Object obj) {
        if (obj instanceof Track) {
            Track track = (Track) obj;
            if (this.titleTrack.equals(track.titleTrack) && this.length == track.length) {
                return true;
            }
        }
        return false;
    }
}
```

Figure 3.4: Track Class

3.3 Open the CompactDisc class



```
Lab04 > AimsProject > J CompactDisc.java
1  import java.util.ArrayList;
2
3  public class CompactDisc extends Disc implements Playable {
4      private String artist;
5      private ArrayList<Track> tracks = new ArrayList<Track>();
6
7      public CompactDisc(int id, String title, String category, float cost) {
8          super(id, title, category, cost);
9      }
10     public CompactDisc(int id, String title, String category, float cost, String artist, ArrayList<Track> tracks) {
11         super(id, title, category, cost);
12         this.artist = artist;
13         this.tracks = tracks;
14     }
15     public void addTrack(Track track) {
16         if(!tracks.contains(track)) {
17             tracks.add(track);
18             System.out.println(track+" has been added");
19         }
20         System.out.println(track+" is already in the track list");
21     }
22     public void removeTrack(Track track) {
23         if(!tracks.contains(track)) {
24             System.out.println(track+" is not in the track list");
25         }
26         else{
27             tracks.remove(track);
28             System.out.println(track+" has been removed from the track list");
29         }
30     }
31     public int getLength() {
32         int totalLength = 0;
33         for (Track track : tracks) {
34             totalLength += track.getLength();
35         }
36         return totalLength;
37     }
38     public String getArtist() {
39         return artist;
40     }
41     public void setArtist(String artist) {
42         this.artist = artist;
43     }
44
45     public String toString() {
46         return "ID - "+getId()+ ". CD - " + getTitle() + " - " + getCategory() + " - " + getDirector() + " - " + getArtist() + " - " + getLength() + "s - $" + getCost();
47     }
48 }
```

Figure 3.5: CompactDisc Class 1

4 Create the Playable interface

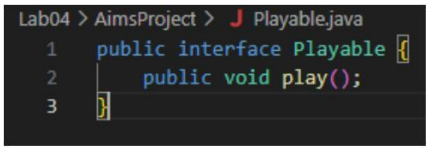


Figure 4.1: Playable interface

Implement play() cho các class DigitalVideoDisc, Track, CompactDisc

```
public void play() {
    System.out.println("Playing DVD: " + this.getTitle());
    System.out.println("DVD length: " + this.getLength());
}
```

Figure 4.2: Method play() của DigitalVideoDisc

```
11  public void play() {
12      System.out.println("Playing DVD: " + this.getTitle());
13      System.out.println("DVD length: " + this.getLength());
14  }
15
```

Figure 4.3: Method play() của Track

```
15  public void play() {
16      System.out.println("Playing CD: " + this.getTitle());
17      System.out.println("CD length: " + this.getLength());
18      for (Track track : tracks) {
19          track.play();
20      }
21  }
```

Figure 4.4: Method play() của CompactDisc

5 Update the Cart class to work with Media

Lớp Cart bây giờ cần có khả năng tương tác với các đối tượng DVD, CD và Book. Vì các lớp DVD, CD và Book đều kế thừa từ lớp Media, nên thay vì làm việc trực tiếp với từng lớp con, lớp cart chỉ cần giao tiếp với lớp Media là có thể hoạt động được với tất cả.

Lab04 > AimsProject > Cart.java

```java
import java.util.ArrayList;

public class Cart {
    private final int MAX_NUMBER_ORDERED = 20;
    private ArrayList<Media> itemsOrdered = new ArrayList<Media>(MAX_NUMBER_ORDERED);

    public void addMedia(Media media) {
        if (itemsOrdered.size()<MAX_NUMBER_ORDERED) {
            itemsOrdered.add(media);
            System.out.println(media.getTitle()+" has been added to the cart.");
        }
        else System.out.println("The cart is full.");
    }

    public void removeMedia(Media media) {
        boolean found = false;
        for (Media item : itemsOrdered) {
            if (item.equals(media)) {
                itemsOrdered.remove(item);
                System.out.println(media.getTitle() + " has been removed from the cart.");
                found = true;
                break;
            }
        }
        if (!found) {
            System.out.println(media.getTitle() + " is not in the cart.");
        }
    }

    public ArrayList<Media> getItemsInCart(){
        return itemsOrdered;
    }

    //Get total cost
    public float totalCost() {
        float total = 0.0f;
        for (int i = 0; i < itemsOrdered.size(); i++) {
            total += itemsOrdered.get(i).getCost();
        }
        return total;
    }

    //Print cart method
    public void printCart() {
        System.out.println("**********************CART**********************");
        System.out.println("Ordered Items:");
        for (int i = 0 ; i < itemsOrdered.size(); i++) {
            System.out.println((i+1) + itemsOrdered.get(i).toString());
```

Figure 5.1: Cart Class 1

```java
    //Print cart method
    public void printCart() {
        System.out.println("**********CART**********");
        System.out.println("Ordered Items:");
        for (int i = 0 ; i < itemsOrdered.size(); i++) {
            System.out.println((i+1) + itemsOrdered.get(i).toString());
        }
        System.out.println("Total cost: $" + totalCost());
        System.out.println("**************************************************");
    }

    //Search for DVDs in the cart by ID and display them
    //Notify to user if no match is found
    public void searchID(int id){
        boolean found = false;
        for (int i = 0; i < itemsOrdered.size(); i++) {
            if (itemsOrdered.get(i).getId() == id) {
                found = true;
                System.out.println("DVD found: ");
                System.out.println(itemsOrdered.get(i).toString());
                break;
            }
        }
        if (!found) System.out.println("No match is found!");
    }

    //Search for DVDs in the cart by title and display them
    //Notify to user if no match is found
    public void searchTitle(String title){
        boolean found = false;
        for (int i = 0; i < itemsOrdered.size(); i++) {
            if (itemsOrdered.get(i).getTitle().equals(title)) {
                found = true;
                System.out.println("DVD found: ");
                System.out.println(itemsOrdered.get(i).toString());
                break;
            }
        }
        if (!found) System.out.println("No match is found!");
    }
}
```

Figure 5.2: Cart Class 2

6 Update the Store class to work with Media

Lab04 > AimsProject > Store.java

```java
import java.util.ArrayList;

import hust.soict.hedspi.aims.media.*;

public class Store {
    private ArrayList<Media> itemsInStore = new ArrayList<Media>();

    public void addMedia(Media media) {
        itemsInStore.add(media);
        System.out.println(media.getTitle()+ " has been added to the store.");
    }

    public void removeMedia(Media media) {
        boolean found = false;
        for (Media item : itemsInStore) {
            if (item.equals(media)) {
                itemsInStore.remove(item);
                System.out.println(media.getTitle() + " has been removed from the store.");
                found = true;
                break;
            }
        }
        if (!found) System.out.println(media.getTitle() + " is not in the store.");
    }
    public ArrayList<Media> getItemsInStore() {return itemsInStore;}
}
```

Figure 6.1: Store Class 1

7 Constructors of whole classes and parent classes

```java
public Track(int id, String title, String category, float cost, String titleTrack, int length) {
    super(id, title, category, cost);
    this.titleTrack = titleTrack;
    this.length = length;
}
```

Figure 7.1: Constructor Track Class

```java
public CompactDisc(int id, String title, String category, float cost) {
    super(id, title, category, cost);
}
public CompactDisc(int id, String title, String category, float cost, String artist, ArrayList<Track> tracks) {
    super(id, title, category, cost);
    this.artist = artist;
    this.tracks = tracks;
}
```

Figure 7.2: Constructor CompactDisc Class

Lớp Disc kế thừa lớp Media, khi đó lớp Media là lớp cha, lớp Disc là lớp con.

```java
public Media(int id, String title, String category, float cost) {
    this.id = id;
    this.title = title;
    this.category = category;
    this.cost = cost;
}
```

Figure 7.3: Constructor Media Class

```java
public Disc(int id, String title, String category, float cost) {
    super(id,title, category, cost);
}

public Disc(int id, String title, String category, float cost, int length, String director) {
    super(id, title, category, cost);
    this.length = length;
    this.director = director;
}
```

Figure 7.4: Constructor Disc Class

8 Unique item in a list

Để tránh trùng lặp các phần tử media trong giỏ hàng hoặc các track trong một đĩa CD, chúng ta có thể ghi đè lại phương thức equals() mặc định kế thừa từ lớp Object. Việc này cho phép so sánh bản chất thay vì so sánh vị trí ô nhớ của các đối tượng, qua đó ngăn chặn thêm các phần tử bị trùng lắp vào danh sách.

```java
public boolean equals(Object obj) {
    if (obj instanceof Media) {
        Media media = (Media) obj;
        if (this.id == media.id) {
            return true;
        }
    }
    return false;
}
```

Figure 8.1: Override equals in Media Class

```java
public boolean equals(Object obj) {
    if (obj instanceof Track) {
        Track track = (Track) obj;
        if (this.titleTrack.equals(track.titleTrack) && this.length == track.length) {
            return true;
        }
    }
    return false;
}
```

Figure 8.2: Override equals in Track Class

9 Polymorphism with toString() method

```java
public static void main(String[] args) {
    ArrayList<Media> media = new ArrayList<>();

    CompactDisc cd = new CompactDisc(1,"Soledad", "Ballad", 12.5f);
    DigitalVideoDisc dvd = new DigitalVideoDisc(3,"Final Fantasy X", "Fantasy", 222.22f );
    Book book = new Book(2,"Operating System Concepts", "ICT", 30f);

    // Add some media objects to the list
    media.add(cd);
    media.add(dvd);
    media.add(book);

    Collections.sort(media, Media.COMPARE_BY_TITLE_COST);
```

Figure 9.1: Code mô phỏng Polymorphism

```java
for (Media media : media) {
    System.out.println(media.toString());
}
```

Figure 9.2: toString() in Media Class

Kết quả

```
- 1. CD - Soledad - Ballad - null - null - 0s - $12.5
- 3. DVD - Final Fantasy X - Fantasy - null - 0s - $222.22
- 2. Book - Operating System Concepts - ICT - [] - $30.0
```

Figure 9.3: Result demo Polymorphism

Lớp Media là lớp cơ sở được kế thừa bởi các lớp cụ thể hơn là CompactDisc, DigitalVideoDisc và Book. Khi khởi tạo các đối tượng cd, dvd, book thuộc lớp con rồi gán chúng cho biến kiểu Media, ta áp dụng kỹ thuật gọi là upcasting.

Việc thêm chúng vào danh sách media và duyệt danh sách để in ra thông tin mỗi phần tử bằng phương thức toString() là ví dụ điển hình cho tính đa hình động. Mỗi lớp con có thể cài đặt riêng toString() nên kết quả sẽ khác nhau dựa theo loại đối tượng, mà không cần quan tâm đến kiểu cụ thể của từng phần tử.

10 Sort media in the cart

Sắp xếp các media trong giỏ hàng theo hai tiêu chí:

- Bằng title: Hiển thị tất cả các media theo thứ tự bảng chữ cái. Trong trường hợp cùng title, media có cost cao hơn sẽ được hiển thị trước.
- Bằng cost: Hiển thị theo thứ tự cost giảm dần. Trong trường hợp cost như nhau, sắp xếp media theo thứ tự bảng chữ cái

```
public static final Comparator<Media> COMPARE_BY_TITLE_COST =
        new MediaComparatorByTitleCost();
public static final Comparator<Media> COMPARE_BY_COST_TITLE =
        new MediaComparatorByCostTitle();
```

Figure 10.1: Add the comparators as attributes of the Media class

```
1  import java.util.Comparator;
2
3  public class MediaComparatorByCostTitle implements Comparator<Media> {
4      public int compare(Media media1, Media media2) {
5          return Comparator.comparing(Media::getCost)
6                  .thenComparing(Media::getTitle)
7                  .compare(media1, media2);
8      }
9  }
```

Figure 10.2: MediaComparatorByCostTitle Class

```java
import java.util.Comparator;

public class MediaComparatorByTitleCost implements Comparator<Media> {
    public int compare(Media media1, Media media2) {
        return Comparator.comparing(Media::getTitle)
                .thenComparing(Media::getCost)
                .compare(media1, media2);
    }
}
```

Figure 10.3: MediaComparatorByTitleCost Class

11 Create a complete console application in the Aims class

```java
public static void showMenu() {
        System.out.println("AIMS: ");
        System.out.println("--------------------------------");
        System.out.println("1.View store");
        System.out.println("2.Update store");
        System.out.println("3.See current cart");
        System.out.println("0. Exit");
        System.out.println("--------------------------------");
        System.out.println("Please choose a number: 0-1-2-3");
}
```

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

Figure 11.1: Màn hình chính

12 Class Diagram

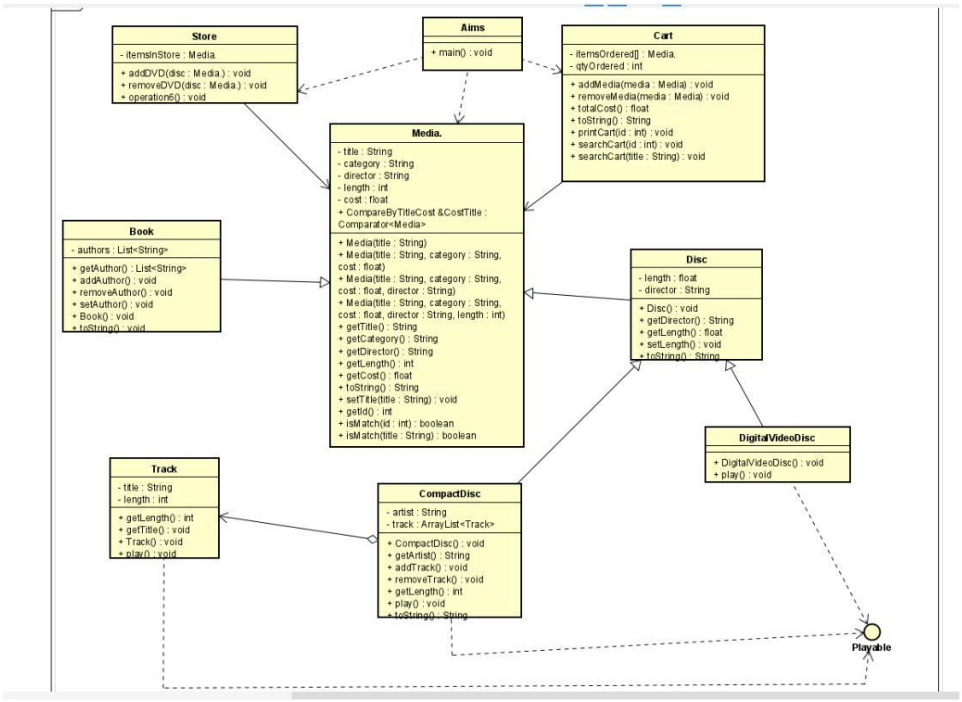


Figure 12.1: Class Diagram

13 UseCase Diagram

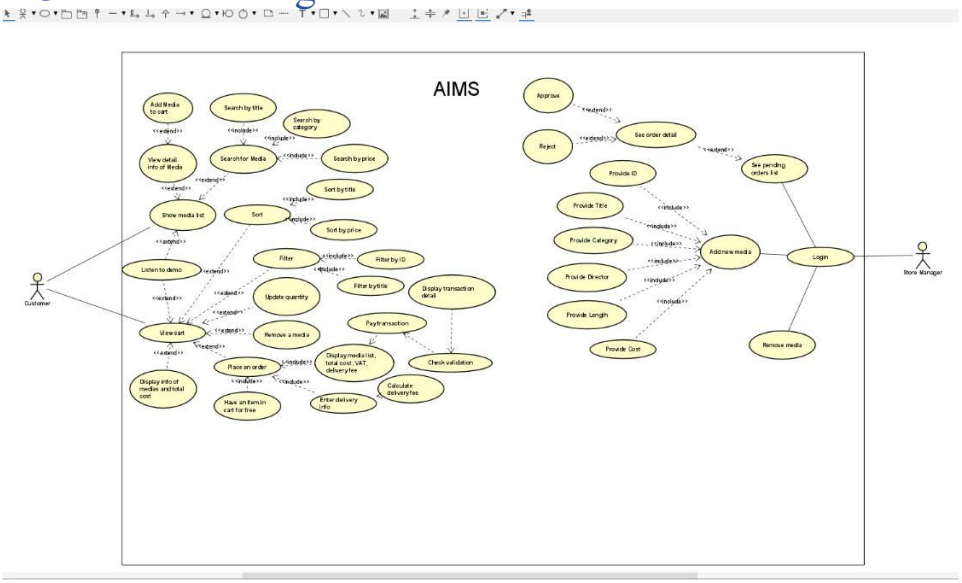


Figure 13.1: UseCase Diagram

14 Answer Questions

- Media class (abstract class) that contains compare object) should implement.
- We should compare objects' attributes.
- No, because compareTo() method only returns one int value.
- Code:

```
public class DVD extends Media implements Comparable {
        public int compareTo(DVD other) {
                if (!this.getTitle().compareTo(other.getTitle());
                }
                else if (this instanceof DVD && other instance of DVD) {
                        return Interger.compare(((DVD) other).getLength(),
((DVD)this).getLength());

                }
                else return Double.compare(this.getCost(), other.getCost());

        }
```