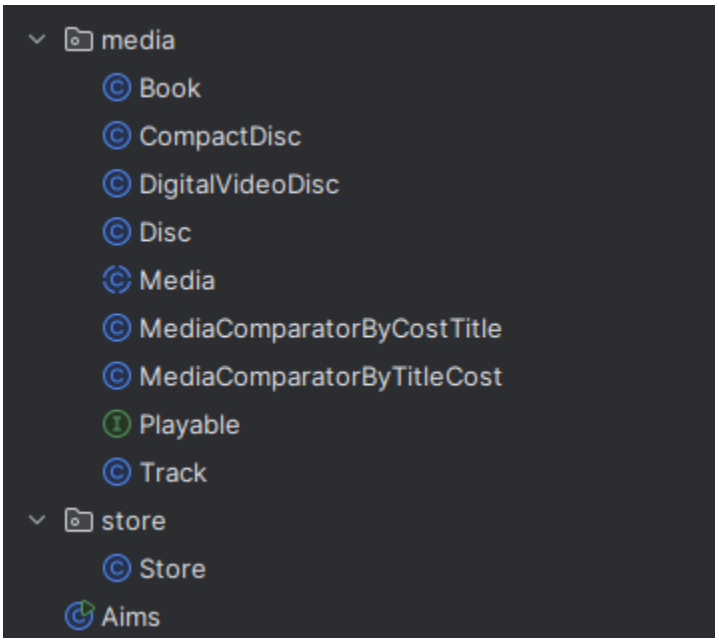
OOP_LAB_04

Họ và tên: Phan Đức Duy

MSSV: 20225831

Mã Lớp: 744520

1. **Import the existing project into the workspace of Eclipse**
2. **Additional requirements of AIMS**



3. **Creating the Book class**

```java
package host.solct.nedspi.dims.media;

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

public class Book extends Media {  16 usages  pdd04

    private List<String> author = new ArrayList<String>();  8 usages

    public Book( String title, String category, float cost, List<String> author) {  6 usages  pdd04
        super(title, category, cost);
        this.author = author;
        nb++;
        setId(nb);
    }

    public Book( String title, String category, float cost) {  1 usage  pdd04
        super(title, category, cost);
        nb++;
        setId(nb);
    }

    public List<String> getAuthor() { return author; }

    public void setAuthor(List<String> author) { this.author = author; }

    public void addAuthor(String author) {  8 usages  pdd04
        if(!this.author.contains(author)){
            this.author.add(author);
        }else{
            System.out.println("Author already exists");
        }
    }
    public void removeAuthor(String author) {  no usages  pdd04
         if(this.author.contains(author)){
             this.author.remove(author);
         }else{
             System.out.println("Author does not exist");
         }
    }
```

4. **Creating the abstract Media class**

```java
public abstract class Media {  48 usages  4 inheritors  pdd04
    private int id;  3 usages
    private String title;  6 usages
    private String category;  4 usages
    private float cost = -1;  4 usages
    protected static int nb =0;  16 usages
    public static final Comparator<Media> COMPARE_BY_TITLE_COST = new MediaComparatorByTitleCost();  1 usage
    public static final Comparator<Media> COMPARE_BY_COST_TITLE = new MediaComparatorByCostTitle();  1 usage

    public Media() {}  1 usage  pdd04

    public Media(int id, String title, String category, float cost) {  no usages  pdd04
        this.id = id;
        this.title = title;
        this.category = category;
        this.cost = cost;
    }

    public Media(String title, String category, float cost) {  4 usages  pdd04
        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; }

    public String getCategory() { return category; }

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

    public float getCost() { return cost; }

    public void setCost(float cost) {  no usages  pdd04
```

5. **Creating the CompactDisc class**

   5.1. Create the Disc class extending the Media class

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

public class Disc extends Media {  2 usages  2 inheritors  pdd04
    private String director;  2 usages
    private int length = -1;  2 usages


    public Disc(String title, String category, float cost, String director, int length) {  5 usages  pdd04
        super(title, category, cost);
        this.director = director;
        this.length = length;
    }

    public Disc(){}  no usages  pdd04

    public Disc(String title, String category, float cost) {  1 usage  pdd04
        super(title, category, cost);
    }

    public String getDirector() {  2 usages  pdd04
        return director;
    }

    public int getLength() {  4 usages  1 override  pdd04
        return length;
    }

    public void setLength(int length) {}  1 usage  pdd04


}
```

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

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

import java.util.Objects;

public class Track implements Playable{  40 usages  pdd04
    private String title;  5 usages
    private int length = -1;  6 usages

    public Track(String title, int length) {  31 usages  pdd04
        this.title = title;
        this.length = length;
    }

    public String getTitle() {  pdd04
        return title;
    }

    public int getLength() {  2 usages  pdd04
        return length;
    }

    public void setLength(int length) {  no usages  pdd04
        this.length = length;
    }

    public void play() {  7 usages  pdd04
        System.out.println("Playing DVD: " + this.getTitle());
        System.out.println("DVD length: " + this.getLength());
    }
```

5.3. Open the CompactDisc class

```java
public class CompactDisc extends Disc implements Playable {  25 usages  pdd04
    public CompactDisc(String artist, String title, String category, float cost, String director) {  1 usage  pdd04
        this.artist = artist;
        updateLength();
        nb++;
        setId(nb);
    }

    public String getArtist() {  no usages  pdd04
        return artist;
    }

    public void addTrack(Track track) {  31 usages  pdd04
        if(!tracks.contains(track)){
            tracks.add(track);
            updateLength();
        }else{
            System.out.println("Track already exists");
        }
    }

    public void removeTrack(Track track) {  no usages  pdd04
        if(tracks.contains(track)){
            tracks.remove(track);
            updateLength();
        }else{
            System.out.println("Track does not exist");
        }
    }

    public int getLength(){  4 usages  pdd04
        int length = super.getLength();
        return length;
    }

    public void updateLength(){  4 usages  pdd04
        int length = 0;
        for(Track track : tracks){
            length += track.getLength();
        }
        super.setLength(length);
    }
```

6. Create the Playable interface

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

public interface Playable {  3 usages  3 implementations  pdd04
    public void play();  1 usage  3 implementations  pdd04
}
```

```java
public class CompactDisc extends Disc implements Playable {  25 usages  pdd04
    public void removeTrack(Track track) {  no usages  pdd04
    }

    public int getLength(){  4 usages  pdd04
        int length = super.getLength();
        return length;
    }

    public void updateLength(){  4 usages  pdd04
        int length = 0;
        for(Track track : tracks){
            length += track.getLength();
        }
        super.setLength(length);
    }

    public void play(){  7 usages  pdd04
        for(Track track : tracks){
            track.play();
        }
    }
```

```java
public class DigitalVideoDisc extends Disc implements Playable {  32 usages  pdd04
    public DigitalVideoDisc(String title) {  3 usages  pdd04
    }

    public DigitalVideoDisc(String title, String category, float cost) {  no usages  pdd04
        super(title, category, cost);
        nb++;
        super.setId(nb);
    }

    public DigitalVideoDisc(String title, String category, String director, float cost) {  no usages  p
        super(title, category, cost, director, length: -1);
        nb++;
        super.setId(nb);
    }

    public DigitalVideoDisc(String title, String category, String director, int length, float cost) {
        super(title, category, cost, director, length);
        nb++;
        super.setId(nb);
    }

    public void play() {  7 usages  pdd04
        System.out.println("Playing DVD: " + this.getTitle());
        System.out.println("DVD length: " + this.getLength());
    }
```

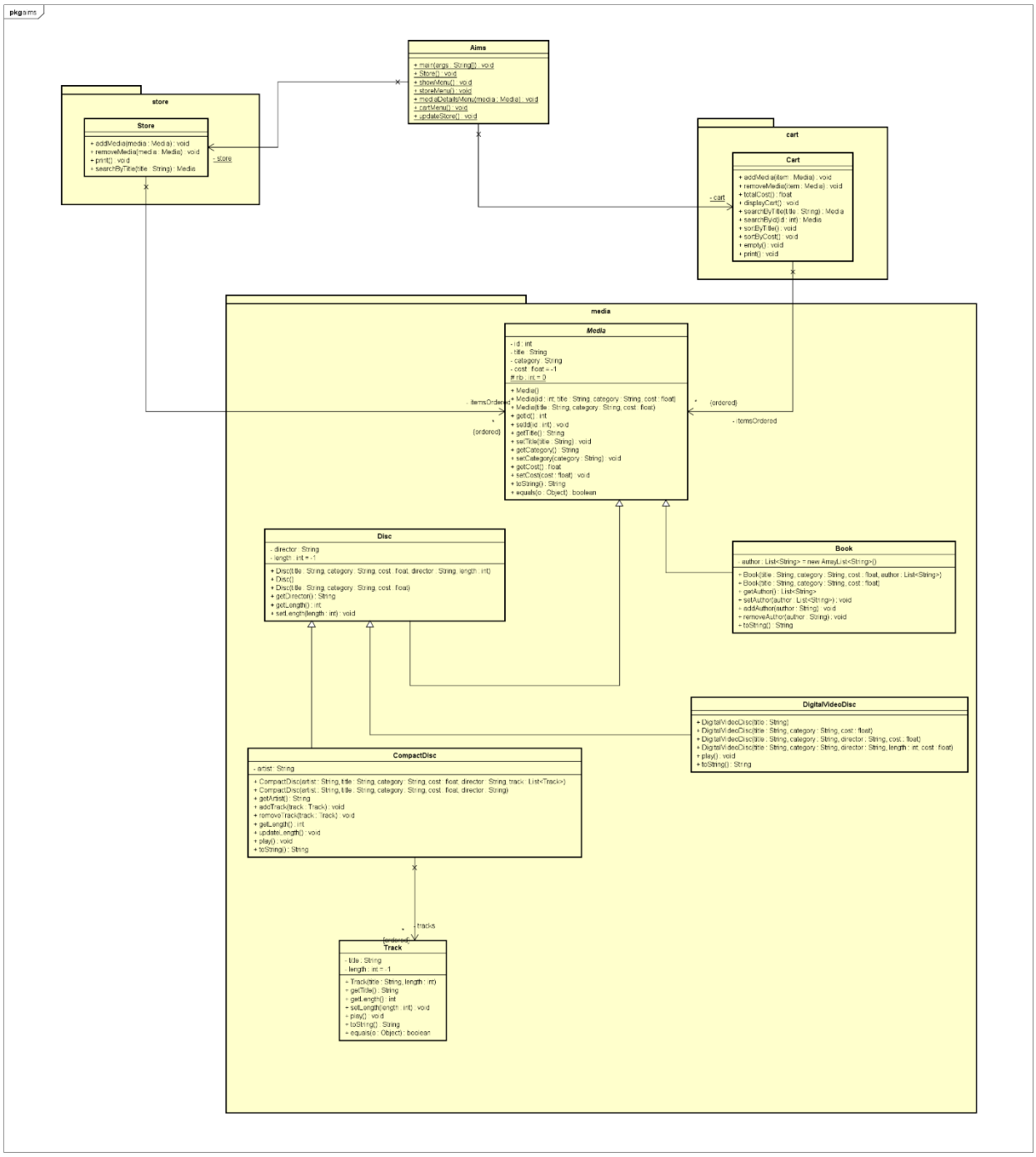
7. **Update the Cart class to work with Media**

```java
public class Cart {  3 usages  pdd04 *

    private ArrayList<Media> itemsOrdered = new ArrayList<Media>();  13 usages

    public void addMedia(Media item) {  2 usages  pdd04
        if(!itemsOrdered.contains(item)) {
            itemsOrdered.add(item);
        }else{
            System.out.println("Duplicate media found");
        }
    }

    public void removeMedia(Media item) {  1 usage  pdd04
        if(itemsOrdered.contains(item)) {
            itemsOrdered.remove(item);
        }else{
            System.out.println("Not found media found");
        }
    }

    public float totalCost() {  1 usage  pdd04
        float total = 0;
        for (Media item : itemsOrdered) {
            total += item.getCost();
        }
        return total;
    }

    public Media searchByTitle(String title) {  3 usages  pdd04
        for (Media media : itemsOrdered) {
            if (media.getTitle().equals(title))
                return media;
        }

        return null;
    }

    public Media searchById(int id) {  1 usage  pdd04
        for (Media media : itemsOrdered) {
            if (media.getId() == id)
                return media;
        }
```

8. **Update the Store class to work with Media**

```java
public class Store {  3 usages  pdd04
    private ArrayList<Media> itemsOrdered = new ArrayList<Media>();  8 usages

    public void addMedia(Media media) {  22 usages  pdd04
        for (Media item : itemsOrdered) {
            if (item.getTitle() == media.getTitle()) {
                System.out.println("DVD already exists!");
                return;
            }
        }
        itemsOrdered.add(media);
        System.out.println("The disc has been added");


    }

    public void removeMedia(Media media) {  1 usage  pdd04
        for (Media item : itemsOrdered) {
            if (item.getTitle().equals(media.getTitle())) {
                itemsOrdered.remove(media);
                System.out.println("The disc has been removed");
                return;
            }
        }
        System.out.println("DVD not found!");
    }

    public void print(){  5 usages  pdd04
        System.out.println("Digital Video Disc:");
        for (Media item : itemsOrdered){
            if(item.getClass() == DigitalVideoDisc.class){
                System.out.println(item.toString());
            }
        }
        System.out.println("CompactDisc:");
        for (Media item : itemsOrdered){
            if(item.getClass() == CompactDisc.class){
                System.out.println(item.toString());
            }
        }
        System.out.println("Book:");
```

9. **Constructors of whole classes and parent classes**



9.1. Câu hỏi: Which classes are aggregates of other classes? Checking all constructors of whole classes if they initialize for their parts?

Trả lời:

- **Cart và Store chứa danh sách các đối tượng Media thông qua ArrayList<Media> itemsOrdered, do đó**
  đây là 2 lớp Tổng hợp.Chúng chứa và quản lý các đối tượng Media trong danh sách. Nếu Cart hoặc Store bị hủy, danh sách Media mà chúng chứa cũng sẽ không còn ý nghĩa.

- **Cart và Store đã khởi tạo các thành phần của chúng (ArrayList<Media>) một cách rõ ràng tại điểm khai báo.**
  Điều này đảm bảo constructor không cần thực hiện thêm thao tác khởi tạo nào.

10. **Unique item in a list**

```
public abstract class Media {  47 usages  4 inheritors  pdd04
    public String toString() {
    }

    @Override  pdd04
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Media media = (Media) o;
        return Objects.equals(title, media.title);
    }
```

```java
public class Track implements Playable{  40 usages  pdd04
        this.title = title;
        this.length = length;
    }

    public String getTitle() {  pdd04
        return title;
    }

    public int getLength() {  2 usages  pdd04
        return length;
    }

    public void setLength(int length) {  no usages  pdd04
        this.length = length;
    }

    public void play() {  7 usages  pdd04
        System.out.println("Playing DVD: " + this.getTitle());
        System.out.println("DVD length: " + this.getLength());
    }

    @Override  pdd04
    public String toString() {
        return "Track: " +
                "title = '" + title + '\'' +
                " - length = " + length + "\n";
    }

    @Override  pdd04
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Track track = (Track) o;
        return length == track.length && Objects.equals(title, track.title);
    }
}
```

10.1. Câu hỏi: When overriding the equals() method of the Object class, you will have to cast the Object parameter obj to the type of Object that you are dealing with. For example, in the Media class, you must cast the Object obj to a Media, and then check the equality of the two objects' attributes as the above requirements (i.e. title for Media; title and length for Track). If the passing object is not an instance of Media, what happens?

Trả lời:

- Nếu đối tượng truyền vào không phải là một instance của Media thì sẽ không thực hiện được phép so sánh.

Nhưng nếu đối tượng truyền là instance con của Media thì ta vẫn sẽ thực hiện được phép so sánh.

11. Polymorphism with toString() method

```
public static void main (String[] args) {  pdd04 *
    List<Media> medias = new ArrayList<>();
    List<Track> tracks = new ArrayList<>();

    medias.add(new CompactDisc( artist: "abc",  title: "song",  category: "remix",  cost: 12,  director: "bca",  tracks));
    medias.add(new DigitalVideoDisc( title: "abc",  category: "phim hai",  cost: 2));
    medias.add(new Book( title: "giai tich", category: "toan cao cap",  cost: 2, new ArrayList<String>()));

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

Kết quả:

```
C:\Users\phand\.jdks\openjdk-22.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2024.1.2\lib\idea_rt.jar
id = 1 - title = 'song' - category = 'remix' - length = -1 - director = 'bca' - artist = 'abc' - cost = 12.0 - tracks = []
id = 2 - title = 'abc' - category = 'phim hai' - length = -1 - director = 'null' - cost = 2.0
id = 3 - title = 'giai tich' - category = 'toan cao cap' - (List of) authors = [] - cost = 2.0

Process finished with exit code 0
```

11.1. Câu hỏi: Iterate through the list and print out the information of the media by using toString() method. Observe what happens and explain in detail.

Trả lời:

Mặc dù trong class Media cũng đã có phương thức toString() nhằm hiển thị ra các thuộc tính và giá trị của các thuộc tính,

nhưng toString() cũng được override để hiển thị chi tiết các thuộc tính hơn trong các lớp CD, DVD và Book.

Do đó java sẽ sử dụng toString() trong các lớp này.

12. Sort media in the cart

```
import java.util.Comparator;

public class MediaComparatorByCostTitle implements Comparator<Media> {  1 usage  pdd04
    @Override  pdd04
    public int compare(Media m1, Media m2) {
        int costComparison = Float.compare(m2.getCost(), m1.getCost());
        if (costComparison != 0) {
            return costComparison;
        }
        return m1.getTitle().compareTo(m2.getTitle());
    }
}
```

```java
public class MediaComparatorByTitleCost implements Comparator<Media> {  no usages
    @Override
    public int compare(Media m1, Media m2) {
        int titleComparison = m1.getTitle().compareTo(m2.getTitle());
        if (titleComparison != 0) {
            return titleComparison;
        }
        return Float.compare(m2.getCost(), m1.getCost());
    }
}
```

```java
public abstract class Media {  4 inheritors  pdd04 *
    private int id;  3 usages
    private String title;  6 usages
    private String category;  4 usages
    private float cost = -1;  4 usages
    protected static int nb =0;  16 usages
    public static final Comparator<Media> COMPARE_BY_TITLE_COST = new MediaComparatorByTitleCost();  2 usages
    public static final Comparator<Media> COMPARE_BY_COST_TITLE = new MediaComparatorByCostTitle();  2 usages

    public Media() {}  1 usage  pdd04
```

Code chạy thử:

```java
public static void main (String[] args) {  new *
    List<Media> medias = new ArrayList<>();
    List<Track> tracks = new ArrayList<>();

    medias.add(new CompactDisc( artist: "abc",  title: "song",  category: "remix",  cost: 12,  director: "bca",  tracks));
    medias.add(new DigitalVideoDisc( title: "abc",  category: "phim hai",  cost: 2));
    medias.add(new Book( title: "giai tich", category: "toan cao cap",  cost: 2, new ArrayList<String>()));

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

    System.out.println("sort by title");
    Collections.sort(medias, COMPARE_BY_TITLE_COST);
    for (Media media : medias) {
        System.out.println(media.toString());
    }

    System.out.println("sort by cost");
    Collections.sort(medias, COMPARE_BY_COST_TITLE);
    for (Media media : medias) {
        System.out.println(media.toString());
    }


}
```

Kết quả:

```
id = 1 - title = 'song' - category = 'remix' - length = -1 - director = 'bca' - artist = 'abc' - cost = 12.0 - tracks = []
id = 2 - title = 'abc' - category = 'phim hai' - length = -1 - director = 'null' - cost = 2.0
id = 3 - title = 'giai tich' - category = 'toan cao cap' - (List of) authors = [] - cost = 2.0
sort by title
id = 2 - title = 'abc' - category = 'phim hai' - length = -1 - director = 'null' - cost = 2.0
id = 3 - title = 'giai tich' - category = 'toan cao cap' - (List of) authors = [] - cost = 2.0
id = 1 - title = 'song' - category = 'remix' - length = -1 - director = 'bca' - artist = 'abc' - cost = 12.0 - tracks = []
sort by cost
id = 1 - title = 'song' - category = 'remix' - length = -1 - director = 'bca' - artist = 'abc' - cost = 12.0 - tracks = []
id = 2 - title = 'abc' - category = 'phim hai' - length = -1 - director = 'null' - cost = 2.0
id = 3 - title = 'giai tich' - category = 'toan cao cap' - (List of) authors = [] - cost = 2.0

Process finished with exit code 0
```

12.1. Câu hỏi: What class should implement the Comparable interface?

- **Lớp chứa đối tượng cần so sánh, chẳng hạn Media hoặc các lớp con của nó như DigitalVideoDisc, Book, CompactDisc.**

12.2. In those classes, how should you implement the compareTo()method be to reflect the ordering that we want?

Triển khai phương thức compareTo() trong lớp Media:

```
@Override
public int compareTo(Media other) {
    int titleComparison = this.title.compareTo(other.title);
    if (titleComparison != 0) return titleComparison;


    return Float.compare(other.cost, this.cost);
}
```

12.3. Câu hỏi: Can we have two ordering rules of the item (by title then cost and by cost then title) if we use this Comparable interface approach?

- **Không ta không thể, Comparable chỉ cho phép định nghĩa một quy tắc sắp xếp duy nhất thông qua phương thức compareTo().**
- **Nếu cần nhiều quy tắc thì khi đó ta phải sử dụng Comparator như đã triển khai trong MediaComparatorByCostTitle và MediaComparatorByTitleCost.**

12.4. Câu hỏi: Suppose the DVDs has a different ordering rule from the other media types, that is by title, then decreasing length, then cost. How would you modify your code to allow this?

Ghi đè phương thức compareTo() trong lớp DigitalVideoDisc:

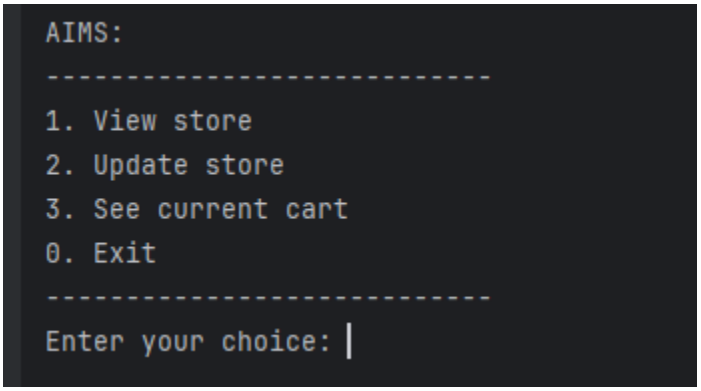
```
public int compareTo(Media other) {
    if (!(other instanceof DigitalVideoDisc)) return super.compareTo(other);

    DigitalVideoDisc otherDVD = (DigitalVideoDisc) other;

    int titleComparison = this.getTitle().compareTo(other.getTitle());
    if (titleComparison != 0) return titleComparison;
```

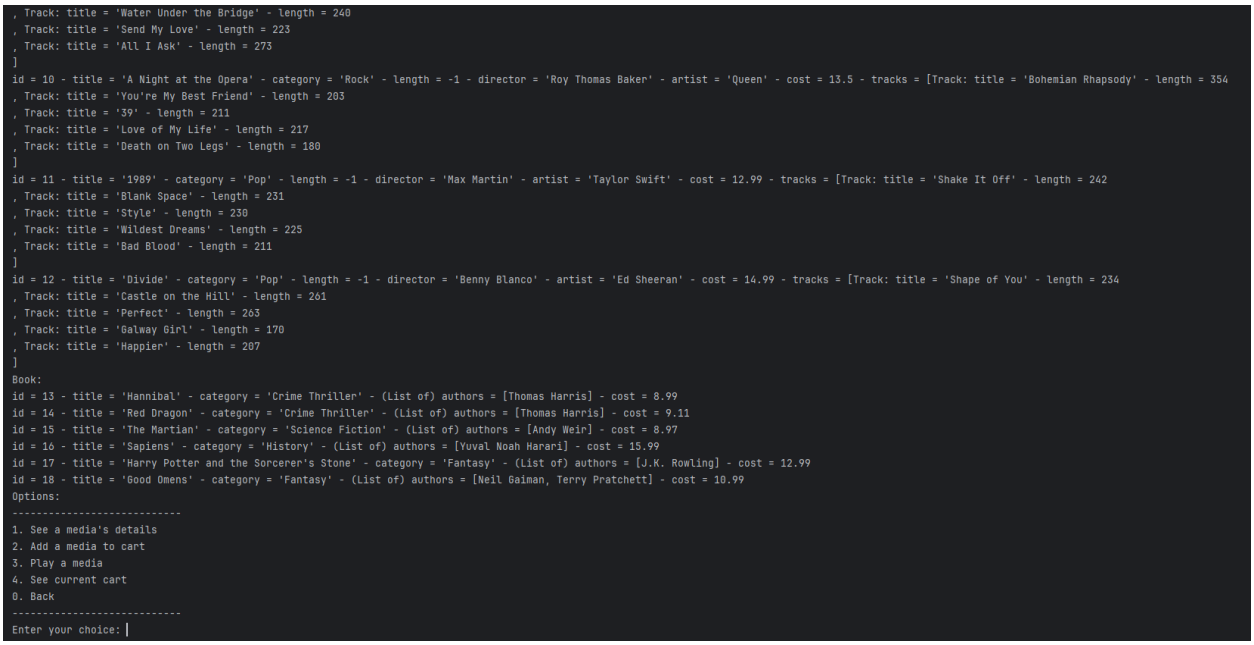
```
        if (this.getLength() == otherDVD.getLength()) return
    Float.compare(otherDVD.getCost(), this.getCost());

        return Integer.compare(otherDVD.getLength(), this.getLength());
    }
```

13. **Create a complete console application in the Aims class**

    Do phần này hơi dài nên e xin phép không chụp code

    Sau đây là một vài kết quả chạy thử:





```
---------------------------
Enter your choice: 2
Playing DVD: Shape of You
DVD length: 234
Playing DVD: Castle on the Hill
DVD length: 261
Playing DVD: Perfect
DVD length: 263
Playing DVD: Galway Girl
DVD length: 170
Playing DVD: Happier
DVD length: 207
Options:
---------------------------
1. Add to cart
2. Play
0. Back
---------------------------
Enter your choice:
```