I. Additional requirements of AIMS

II. Creating the Book class

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

import java.util.ArrayList;

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

    public Book() {
        super();
    }

    public Book(int id, String title, String category, float cost, ArrayList<String> authors) {
        super(title, category, cost);
        this.id = id;
        this.title = title;
        this.category = category;
        this.cost = cost;
        this.authors = authors;
    }

    public Book(String title, String author, float cost) {
        super(title);
        this.title = title;
        this.cost = cost;
```

```
        this.authors.add(author);
    }

    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;
    }
```

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

public void addAuthor(String authorName) {
    if (!authors.contains(authorName)) {
        authors.add(authorName);
        System.out.println("Author added: " + authorName);
    } else {
        System.out.println("Author already exists.");
    }
}

public void removeAuthor(String authorName) {
    if (authors.contains(authorName)) {
        authors.remove(authorName);
        System.out.println("Author removed: " + authorName);
    } else {
        System.out.println("Author not found.");
    }
}

@Override
public String getType() {
    return "Book";
}

@Override
public String getDetails() {
```

```
        StringBuilder details = new StringBuilder();
        details.append("Book Details:\n");
        details.append("ID: ").append(id).append("\n");
        details.append("Title: ").append(title).append("\n");
        details.append("Category: ").append(category).append("\n");
        details.append("Cost: $").append(cost).append("\n");
        details.append("Authors: ").append(authors.isEmpty() ? "None" : String.join(", ",
authors)).append("\n");
        return details.toString();
    }
}
```

III. Creating the abstract Media class

1. Updating book class

```
package hust.soict.dsai.aims.media;

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

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

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

```java
public int getNumPages() {
    return this.numPages;
}


public void addAuthor(String authorName) {
    if (authors.contains(authorName)) {
        System.out.println("Author already on the list!");
        return;
    }
    authors.add(authorName);
    System.out.println("Author " + authorName + " added to the list!");
    return;
}

public void removeAuthor(String authorName) {
    if (authors.contains(authorName)) {
        authors.remove(authorName);
        System.out.println("The author: " + authorName + " removed!");
        return;
    }
    System.out.println("The author: " + authorName + " is not in the lsit!");
    return;
}

public Book(int id, String title, String category, float cost, List<String> author) {
    super(id, title, category, cost);
    this.authors = author;
}
public Book(int id, String title, String category, float cost) {
    super(id,title, category, cost);
```

```
    }


    public Book(String title, String category, int numPages, float cost) {
        super(title, category, cost);
        this.numPages = numPages;
    }
    public Book(String title, String category, float cost)
    {
        super(title, category, cost);
    }
    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("Book Information:\n");
        sb.append("Title: ").append(getTitle()).append("\n");
        sb.append("Category: ").append(getCategory()).append("\n");
        sb.append("Cost: ").append(getCost()).append("\n");
        sb.append("Authors: ").append(getAuthors()).append("\n");
        return sb.toString();
    }
    @Override
    public String getType() {
        return "Book";
    }
    @Override
    public String getDetails() {
        return null;
    }

}
```

2. Updating DigitalVideoDisc class

```
package hust.soict.dsai.aims.media;

public class DigitalVideoDisc extends Disc implements Playable {

    public String getType() {
        return "DVD";
    }

    public String getDetails() {
        return "Product ID: " + this.getId()
                + "\n\tTitle: " + this.getTitle()
                + "\n\tCategory: " + this.getCategory()
                + "\n\tDirector: " + this.getDirector()
                + "\n\tLength: " + this.getLength() + " minutes"
                + "\n\tPrice: $" + this.getCost();
    }

    public void play() {
        if (this.getLength() <= 0) {
            System.out.println("ERROR: DVD length is non-positive!");
        } else {
            System.out.println("Playing DVD: " + this.getTitle());
            System.out.println("DVD length: " + this.getLength() + " minutes");
        }
    }

    public DigitalVideoDisc(String title) {
        super(title);
```

```
    }

    public DigitalVideoDisc(String title, String category, float cost) {
        super(title, category, cost);
    }

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

3. Creating media abstract class

```
package hust.soict.dsai.aims.media;

import java.time.LocalDate;
import java.util.Comparator;
public abstract class Media {
    private int id;
    private String title;
    private String category;
    private LocalDate dateAdded;
    private float cost;

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

```java
public Media() {
}

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

public Media(String title){
    this.setTitle(title);
}

public Media(String title, String category, float cost){
    this(title);
    this.setCategory(category);
    this.setCost(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;
}
```

```java
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;
}

public boolean isMatch(String title) {
    return this.getTitle().equals(title);
}

@Override
public boolean equals(Object obj) {
    if(this==obj) return true;
    if(obj == null || getClass() != obj.getClass()) return false;

    Media media = (Media) obj;
    return title.equals(media.title);
}

public int compareTo(Media obj) throws NullPointerException {
    try {
        for (int i = 0; i < this.title.length() && i < obj.getTitle().length(); i++) {
            if ((int) this.title.charAt(i) == (int) obj.getTitle().charAt(i)) {
```

```
                continue;
            } else {
                return ((int) this.title.charAt(i) - (int) obj.getTitle().charAt(i));
            }
        }
        if (!(this.title.length() == obj.getTitle().length())) {
            return (this.title.length() - obj.getTitle().length());
        }
        for (int i = 0; i < this.category.length() && i < obj.getCategory().length(); i++) {
            if ((int) this.category.charAt(i) == (int) obj.getCategory().charAt(i)) {
                continue;
            } else {
                return ((int) this.category.charAt(i) - (int) obj.getTitle().charAt(i));
            }
        }
        if (!(this.category.length() == obj.getCategory().length())) {
            return (this.category.length() - obj.getCategory().length());
        }
        return 0;
    } catch (NullPointerException e) {
        throw e;
    }
}

public boolean search(String title) {
    return this.title.toLowerCase().contains(title.toLowerCase());
}

public LocalDate getDateAdded() {
    return dateAdded;
```

```
    }

    public void setDateAdded(LocalDate date) {
        this.dateAdded = date;
    }

    public abstract String getType();

    public abstract String getDetails();

    public String toString() {
        return this.getDetails();
    }
}
```

IV. Creating the CompactDisc class

```
package hust.soict.dsai.aims.media;

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

public class CompactDisc extends Disc implements Playable {

    private String artist;
    private int length;
    private List<Track> tracks = new ArrayList<Track>();

    public String getArtist() {
        return artist;
    }
```

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

public List<Track> getTracks() {
    return this.tracks;
}

public void setTracks(List<Track> tracks) {
    this.tracks = tracks;
}

public void addTrack(Track track) {
    if (tracks.contains(track)) {
        System.out.println("This track already exists!");
        return;
    }
    tracks.add(track);
    System.out.println("The track: " + track.getTitle() + " added to list!");
}

public void removeTrack(Track track) {
    if (tracks.contains(track)) {
        tracks.remove(track);
        System.out.println("The track: " + track.getTitle() + " removed from list!");
        return;
    }
    System.out.println("Track not found in the list!");
}
```

```
    public int getLength() {
        int totalLength = 0;
        for (Track track : tracks) {
            totalLength += track.getLength();
        }
        return totalLength;
    }

    // Constructor
    public CompactDisc(int id, String title, String category, float cost, int length, String director,
String artist, List<Track> track) {
        super(String.valueOf(id), title, category, cost);
        this.artist = artist;
        this.tracks = track;
    }

    public CompactDisc(int id, String title, String category, float cost) {
        super(String.valueOf(id), title, category, cost);
    }

    public CompactDisc(String title) {
        super(title);
    }

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

```
}

public CompactDisc(String title, String author, String director, float cost) {
    super(title, author, director, cost);
}

@Override
public void play() {
    if (this.getLength() <= 0) {
        System.out.println("ERROR: CD length is non-positive!");
    } else {
        System.out.println("Playing CD: " + this.getTitle());
        System.out.println("CD length: " + getLength() + " minutes");

        Iterator<Track> iter = tracks.iterator();
        while (iter.hasNext()) {
            Track track = iter.next();
            track.play();
        }
    }
}

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append("Compact Disc Information:\n");
    sb.append("Title: ").append(getTitle()).append("\n");
    sb.append("Category: ").append(getCategory()).append("\n");
    sb.append("Cost: ").append(getCost()).append("\n");
    sb.append("Artist: ").append(artist).append("\n");
```

```
        sb.append("Number of Tracks: ").append(tracks.size()).append("\n");
        sb.append("Total Length: ").append(getLength()).append(" minutes").append("\n");
        return sb.toString();
    }

    @Override
    public String getType() {
        return "Compact Disc";
    }

    @Override
    public String getDetails() {
        return this.toString();
    }
}
```

4.1. Create the Disc class extending the Media class

```
package hust.soict.dsai.aims.media;

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

    public Disc(String title) {
    }
    public Disc(String director, String category, String title, float cost){
        super(category,title,cost);
        this.setDirector(director);
    }
```

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

    public String getDirector() {
        return director;
    }

    public void setDirector(String director) {
        this.director = director;
    }

    public int getLength() {
        return length;
    }

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

}
```

4.2. 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.dsai.aims.media;

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

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

    public String getTitle() {
        return title;
    }

    public int getLength() {
        return length;
    }

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

    @Override
```

```
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Track track = (Track) obj;
        return title.equals(track.title) && length == track.length;
    }
}
```

V. Create the Playable interface

```
package hust.soict.dsai.aims.media;

public interface Playable {
    public void play();
}
```

VI. Update the Cart class to work with Media

```
package hust.soict.dsai.aims.cart.Cart;

import hust.soict.dsai.aims.media.Media;
import java.util.ArrayList;
import java.util.List;

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

    public void addMedia(Media media) {
```

```
        if (this.itemsOrdered.size() < MAX_NUMBERS_ORDERED) {
            if (this.itemsOrdered.contains(media)) {
                System.out.println("The media already exists in the cart.");
            } else {
                this.itemsOrdered.add(media);
                System.out.println("The media has been added to the cart.");
            }
        } else {
            System.out.println("ERROR: The number of media has reached its limit.");
        }
    }

    public void removeMedia(Media medium) {
        if (this.itemsOrdered.remove(medium)) {
            System.out.println(medium.getTitle() + " has been removed from the cart.");
        } else {
            System.out.println(medium.getTitle() + " is not in the cart.");
        }
    }

    public float totalCost() {
        float cost = 0.0f;
        for (Media medium : itemsOrdered) {
            cost += medium.getCost();
        }
        return cost;
    }

    public void print() {
        System.out.println("\n**********CART**********");
```

```
        System.out.println("Ordered Items:");
        for (int i = 0; i < this.itemsOrdered.size(); i++) {
            System.out.println((i + 1) + ".\t" + this.itemsOrdered.get(i).getDetails() + "\n");
        }
        System.out.println("Total cost: $" + this.totalCost());
        System.out.println("**********************************************\n");
    }

    public boolean filterMedia(int id) {
        boolean found = false;
        int qty = 0;
        float cost = 0f;
        System.out.println("\n**********SEARCH
RESULT**********");
        System.out.println("Product ID: " + id);
        for (int i = 0; i < this.itemsOrdered.size(); i++) {
            if (this.itemsOrdered.get(i).getId() == id) {
                System.out.println((i + 1) + ".\t" + this.itemsOrdered.get(i).getDetails() +
"\n");
                qty += 1;
                cost = this.itemsOrdered.get(i).getCost();
                found = true;
            }
        }
        if (found) {
            System.out.println("Total number of product " + id + " found: " + qty);
            System.out.println("Total cost for these products: $" + (cost * qty));
            System.out.println("**********************************************\
n");
            return true;
```

```
        } else {
            System.out.println("Such product is not in the cart.");
            System.out.println("**************************************************\n");
            return false;
        }
    }

    public boolean filterMedia(String title) {
        boolean found = false;
        int qty = 0;
        float cost = 0f;
        System.out.println("\n*****************SEARCH RESULT*********************");
        System.out.println("Product title: " + title);
        for (int i = 0; i < this.itemsOrdered.size(); i++) {
            if (this.itemsOrdered.get(i).search(title)) {
                System.out.println((i + 1) + ".\t" + this.itemsOrdered.get(i).getDetails() + "\n");
                qty += 1;
                cost = this.itemsOrdered.get(i).getCost();
                found = true;
            }
        }
        if (found) {
            System.out.println("Total number of product \"" + title + "\" found: " + qty);
            System.out.println("Total cost for these products: $" + (cost * qty));
            System.out.println("**************************************************\n");
            return true;
```

```
        } else {
            System.out.println("Such product is not in the cart.");
            System.out.println("**********************************************************\
n");
            return false;
        }
    }

    public Media searchMedia(String title) {
        for (Media medium : this.itemsOrdered) {
            if (medium.getTitle().equalsIgnoreCase(title)) {
                return medium;
            }
        }
        return null;
    }

    public void sortByTitle() {
        itemsOrdered.sort(Media.COMPARE_BY_TITLE_COST);
    }

    public void sortByCost() {
        itemsOrdered.sort(Media.COMPARE_BY_COST_TITLE);
    }

    public int getSize() {
        return this.itemsOrdered.size();
    }

    public List<Media> getItemsOrdered() {
```

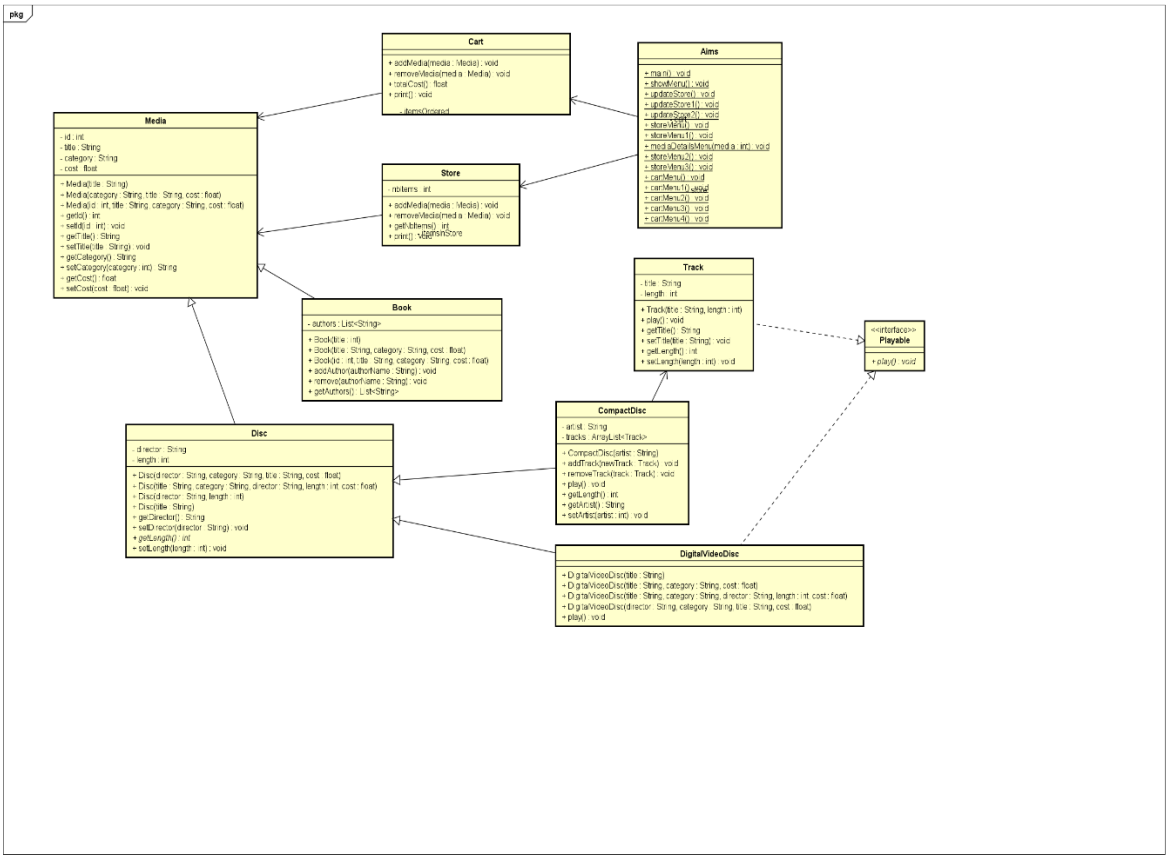
```
        return this.itemsOrdered;
    }

    public List<Media> filterId(String str) {
        List<Media> viewFilter = new ArrayList<>();
        for (Media media : this.itemsOrdered) {
            String idStr = String.valueOf(media.getId());
            if (idStr.startsWith(str)) {
                viewFilter.add(media);
            }
        }
        return viewFilter;
    }

    public List<Media> filterTitle(String str) {
        List<Media> viewFilter = new ArrayList<>();
        for (Media media : this.itemsOrdered) {
            if (media.getTitle().startsWith(str)) {
                viewFilter.add(media);
            }
        }
        return viewFilter;
    }

    public void empty() {
        this.itemsOrdered.clear();
    }
}
```

VII. Update the Store class to work with Media

```java
package hust.soict.dsai.aims.store.Store;

import hust.soict.dsai.aims.media.Media;
import java.util.ArrayList;

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

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

    public void removeMedia(Media media) {
        if (itemsInStore.remove(media)) {
            System.out.println("Media removed from store.");
        } else {
            System.out.println("Media not found in store.");
        }
    }

    public Media searchMedia(String title) {
        for (Media medium : itemsInStore) {
            if (medium.getTitle().equalsIgnoreCase(title)) {
                return medium;
            }
        }
        return null;
    }
```

```
    public void print() {
        System.out.println("\n**********STORE**********"
);
        System.out.println("Items in Store:");
        for (int i = 0; i < itemsInStore.size(); i++) {
            System.out.println((i + 1) + ".\t" + itemsInStore.get(i).getDetails());
        }
        System.out.println("**************************************************\n");
    }
}
```

VIII. Constructors of whole classes and parent classes



IX. Unique item in a list

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

X. Polymorphism with toString() method

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

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

public class Polymorphism {

    public static void main(String[] args) {
        List<Media> media = new ArrayList<Media>();
        CompactDisc cd = new CompactDisc("DVD book","John","Williams",6.00f);
        DigitalVideoDisc dvd = new DigitalVideoDisc("Casas","Rap","Traevis",7,8.52f);
        Book book = new Book("Life of Pi","Adventure",5.32f);
        media.add(cd);
        media.add(dvd);
        media.add(book);

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

```
    }

}
```

XI. Sort media in the cart

```
package hust.soict.dsai.aims.media;

import java.util.Comparator;
public class MediaComparatorByCostTitle implements Comparator<Media>
{
    public MediaComparatorByCostTitle() {
        // TODO Auto-generated constructor stub
    }

    @Override
    public int compare(Media m1, Media m2) {
        if(m1.getCost() != m2.getCost()) {
            return Float.compare(m1.getCost(), m2.getCost());
        }
        return m1.getTitle().compareTo(m2.getTitle());
    }

}
package hust.soict.dsai.aims.media;

import java.util.Comparator;
public class MediaComparatorByTitleCost implements Comparator<Media> {
    public MediaComparatorByTitleCost() {
```

```
    }

    @Override
    public int compare(Media m1, Media m2) {
        if(m1.getTitle().equals(m2.getTitle())) {
            return Float.compare(m1.getCost(),m2.getCost());
        }
        return m1.getTitle().compareTo(m2.getTitle());
    }

}
```

- Question answer in the answers.txt file

XII. Create a complete console application in the Aims class

```
package hust.soict.dsai.aims.AIMS;

import hust.soict.dsai.aims.cart.Cart.Cart;
import hust.soict.dsai.aims.media.*;
import hust.soict.dsai.aims.store.Store.Store;

import java.util.Scanner;

public class Aims {
    private static Store store = new Store();
    private static Cart cart = new Cart();
    private static Scanner scan = new Scanner(System.in);

    public static void main(String[] args) {
```

```
        initStore();
        showMenu();
    }

    public static void showMenu() {
        while (true) {
            System.out.println("\nAIMS: ");
            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");

            int choice = getValidChoice(0, 3);
            switch (choice) {
                case 1:
                    storeMenu();
                    break;
                case 2:
                    updateStoreMenu();
                    break;
                case 3:
                    cartMenu();
                    break;
                case 0:
                    System.out.println("Thank you for using our service. We hope to see you
again.");
                    return;
```

```
        }
    }
}

public static void storeMenu() {
    while (true) {
        System.out.println();
        store.print();
        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 choice = getValidChoice(0, 4);
        switch (choice) {
            case 1:
                seeMediaDetails();
                break;
            case 2:
                addMediaToCart();
                break;
            case 3:
                playMediaInStore();
                break;
            case 4:
```

```
                cartMenu();
                break;
            case 0:
                return;
        }
    }
}

public static void seeMediaDetails() {
    System.out.println("Enter the title of the media:");
    String title = scan.nextLine();
    Media media = store.searchMedia(title);
    if (media != null) {
        System.out.println(media.toString());
        mediaDetailsMenu(media);
    } else {
        System.out.println("Media not found.");
    }
}

public static void mediaDetailsMenu(Media media) {
    while (true) {
        System.out.println("Options: ");
        System.out.println("------------------------------");
        System.out.println("1. Add to cart");
        if (media instanceof Playable) {
            System.out.println("2. Play");
        }
        System.out.println("0. Back");
        System.out.println("------------------------------");
```

```
            if (media instanceof Playable) {
                System.out.println("Please choose a number: 0-1-2");
            } else {
                System.out.println("Please choose a number: 0-1");
            }

            int choice;
            if (media instanceof Playable) {
                choice = getValidChoice(0, 2);
            } else {
                choice = getValidChoice(0, 1);
            }

            switch (choice) {
                case 1:
                    cart.addMedia(media);
                    System.out.println("Media added to cart. Total items in cart: " +
cart.getSize());
                    break;
                case 2:
                    if (media instanceof Playable) {
                        ((Playable) media).play();
                    }
                    break;
                case 0:
                    return;
            }
        }
    }
```

```java
public static void addMediaToCart() {
    System.out.println("Enter the title of the media:");
    String title = scan.nextLine();
    Media media = store.searchMedia(title);
    if (media != null) {
        cart.addMedia(media);
        System.out.println("Media added to cart. Total items in cart: " + cart.getSize());
    } else {
        System.out.println("Media not found.");
    }
}

public static void playMediaInStore() {
    System.out.println("Enter the title of the media:");
    String title = scan.nextLine();
    Media media = store.searchMedia(title);
    if (media != null && media instanceof Playable) {
        ((Playable) media).play();
    } else if (media != null) {
        System.out.println("Media is not playable.");
    } else {
        System.out.println("Media not found.");
    }
}

public static void updateStoreMenu() {
    while (true) {
        System.out.println("Options: ");
        System.out.println("--------------------------------");
        System.out.println("1. Add media to the store");
```

```
            System.out.println("2. Remove media from the store");
            System.out.println("0. Back");
            System.out.println("------------------------------");
            System.out.println("Please choose a number: 0-1-2");

            int choice = getValidChoice(0, 2);
            switch (choice) {
                case 1:
                    addMediaToStore();
                    break;
                case 2:
                    removeMediaFromStore();
                    break;
                case 0:
                    return;
            }
        }
    }

    public static void addMediaToStore() {
        System.out.println("Enter the type of media to add (Book/CD/DVD):");
        String type = scan.nextLine().trim().toLowerCase();
        Media media = null;
        System.out.println("Enter title:");
        String title = scan.nextLine();
        System.out.println("Enter category:");
        String category = scan.nextLine();
        System.out.println("Enter cost:");
        float cost = scan.nextFloat();
        scan.nextLine(); // Consume newline
```

```
        switch (type) {
            case "book":
                media = new Book(title, category, cost);
                break;
            case "cd":
                System.out.println("Enter artist:");
                String artist = scan.nextLine();
                media = new CompactDisc(title, category, artist, artist, 0, cost);
                break;
            case "dvd":
                System.out.println("Enter director:");
                String director = scan.nextLine();
                media = new DigitalVideoDisc(title, category, director, 0, cost);
                break;
            default:
                System.out.println("Invalid media type.");
                return;
        }
        store.addMedia(media);
        System.out.println("Media added to store.");
    }

    public static void removeMediaFromStore() {
        System.out.println("Enter the title of the media to remove:");
        String title = scan.nextLine();
        Media media = store.searchMedia(title);
        if (media != null) {
            store.removeMedia(media);
            System.out.println("Media removed from store.");
```

```
        } else {
            System.out.println("Media not found.");
        }
    }

    public static void cartMenu() {
        while (true) {
            cart.print();
            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 choice = getValidChoice(0, 5);
            switch (choice) {
                case 1:
                    filterCartMenu();
                    break;
                case 2:
                    sortCartMenu();
                    break;
                case 3:
                    removeFromCart();
                    break;
```

```
            case 4:
                playMediaInCart();
                break;
            case 5:
                placeOrder();
                break;
            case 0:
                return;
        }
    }
}

public static void filterCartMenu() {
    System.out.println("Filter by:");
    System.out.println("1. ID");
    System.out.println("2. Title");
    int choice = getValidChoice(1, 2);
    switch (choice) {
        case 1:
            System.out.println("Enter ID:");
            int id = scan.nextInt();
            scan.nextLine();
            cart.filterMedia(id);
            break;
        case 2:
            System.out.println("Enter title:");
            String title = scan.nextLine();
            cart.filterMedia(title);
            break;
    }
```

```java
}

public static void sortCartMenu() {
    System.out.println("Sort by:");
    System.out.println("1. Title");
    System.out.println("2. Cost");
    int choice = getValidChoice(1, 2);
    switch (choice) {
        case 1:
            cart.sortByTitle();
            break;
        case 2:
            cart.sortByCost();
            break;
    }
    cart.print();
}

public static void removeFromCart() {
    System.out.println("Enter the title of the media to remove:");
    String title = scan.nextLine();
    Media media = cart.searchMedia(title);
    if (media != null) {
        cart.removeMedia(media);
        System.out.println("Media removed from cart.");
    } else {
        System.out.println("Media not found in cart.");
    }
}
```

```
    public static void playMediaInCart() {
        System.out.println("Enter the title of the media:");
        String title = scan.nextLine();
        Media media = cart.searchMedia(title);
        if (media != null && media instanceof Playable) {
            ((Playable) media).play();
        } else if (media != null) {
            System.out.println("Media is not playable.");
        } else {
            System.out.println("Media not found in cart.");
        }
    }

    public static void placeOrder() {
        System.out.println("Order placed successfully.");
        cart.empty();
    }

    private static int getValidChoice(int min, int max) {
        int choice = -1;
        while (choice < min || choice > max) {
            if (scan.hasNextInt()) {
                choice = scan.nextInt();
                scan.nextLine();
                if (choice < min || choice > max) {
                    System.out.println("Invalid choice. Please enter a number between " + min
+ " and " + max);
                }
            } else {
```

```
                System.out.println("Invalid input. Please enter a number between " + min + "
and " + max);
                scan.nextLine();
            }
        }
        return choice;
    }

    private static void initStore() {
        store.addMedia(new Book("The Great Gatsby", "Classic", 15.99f));
        store.addMedia(new DigitalVideoDisc("Inception", "Science Fiction", "Christopher
Nolan", 148, 24.99f));
        store.addMedia(new CompactDisc("Thriller", "Pop", "Michael Jackson", "Michael
Jackson", 42, 19.99f));
    }
}
```