Phạm Quang Minh - 20235606

2. Additional requirements of **AIMS**

```java
package hust.soict.dsai.aims.aims;
import java.util.Scanner;

import hust.soict.dsai.aims.cart.Cart;
import hust.soict.dsai.aims.media.Book;
import hust.soict.dsai.aims.media.CompactDisc;
import hust.soict.dsai.aims.media.DigitalVideoDisc;
import hust.soict.dsai.aims.media.Disc;
import hust.soict.dsai.aims.media.Media;
import hust.soict.dsai.aims.media.Track;
import hust.soict.dsai.aims.store.Store;

public class Aims {
    private static Store store = new Store();
    private static Cart cart = new Cart();

    public static void main(String[] args) {

        initSetup();

        boolean exit = false;
        while (!exit) {

            showMenu();

            Scanner scanner = new Scanner(System.in);
            int option = scanner.nextInt();
            scanner.nextLine();

            switch (option) {
                case 0:
                    exit = true;
                    System.out.println("Good bye!");
                    break;
                case 1:
                    clearConsole();
                    storeMenu(scanner);
                    break;
                case 2:
                    clearConsole();
                    updateStoreMenu(scanner);
                    break;
                case 3:
                    clearConsole();
                    cartMenu(scanner);
                    break;
                default:
                    clearConsole();
                    System.out.println("Invalid option, please choose again.");
                    break;
            }

        }

```

```
}
public static void clearConsole() {
    System.out.print(s:"\033[H\033[2J");
    System.out.flush();
    }
// init store setup
public static void initSetup() {

    DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King", "Animation", "Roger Allers", 87, 19.95f);
    DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star War", "Science Fiction", "George Lucas", 87, 24.95f);
    DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin", "Animation", 18.99f);
    store.addMedia(dvd1);
    store.addMedia(dvd2);
    store.addMedia(dvd3);


    Book book1 = new Book("The Silent Patient", "Thriller", 180.00f);
    Book book2 = new Book("Educated: A Memoir", "Biography", 250.00f);
    Book book3 = new Book("The Subtle Art of Not Giving a F*ck", "Self-help", 300.00f);
    store.addMedia(book1);
    store.addMedia(book2);
    store.addMedia(book3);


    CompactDisc cd1 = new CompactDisc("Divide", "Music", "Ed Sheeran", 1200.50f);
    Track track1CD1 = new Track("Shape of You", 263);
    Track track2CD1 = new Track("Castle on the Hill", 261);
    Track track3CD1 = new Track("Perfect", 263);
    cd1.addTrack(track1CD1);
    cd1.addTrack(track2CD1);
    cd1.addTrack(track3CD1);


    CompactDisc cd2 = new CompactDisc("Future Nostalgia", "Music", "Dua Lipa", 1600.75f);
    Track track1CD2 = new Track("Levitating", 203);
    Track track2CD2 = new Track("Don't Start Now", 183);
    Track track3CD2 = new Track("Physical", 191);
    cd2.addTrack(track1CD2);
    cd2.addTrack(track2CD2);
    cd2.addTrack(track3CD2);

    store.addMedia(cd1);
    store.addMedia(cd2);

    clearConsole();
}
```

```java
    // Print method
    public static void showMenu() {
        System.out.println(x:"AIMS: ");
        System.out.println(x:"--------------------------------");
        System.out.println(x:"1. View store");
        System.out.println(x:"2. Update store");
        System.out.println(x:"3. See current cart");
        System.out.println(x:"0. Exit");
        System.out.println(x:"--------------------------------");
        System.out.println(x:"Please choose a number: 0-1-2-3");
    }
    public static void storeMenu(Scanner scanner) {
        boolean back = false;
        while (!back) {
            store.printStore();
            System.out.println(x:"Options: ");
            System.out.println(x:"--------------------------------");
            System.out.println(x:"1. See a media's details");
            System.out.println(x:"2. Add a media to cart");
            System.out.println(x:"3. Play a media");
            System.out.println(x:"4. See current cart");
            System.out.println(x:"0. Back");
            System.out.println(x:"--------------------------------");
            System.out.println(x:"Please choose a number: 0-1-2-3-4");
            int option = scanner.nextInt();
            scanner.nextLine();
            switch (option) {
                case 0:
                    clearConsole();
                    back = true;
                    break;
                case 1:
                    boolean foundDetails = false;
                    while (!foundDetails) {
                        System.out.println(x:"Enter the title of the media (type 0 to stop): ");
                        String title = scanner.nextLine();
                        if (title.equals(anObject:"0")) {
                            clearConsole();
                            break;
                        }
                        Media media = store.search(title);
                        if (media != null) {
                            clearConsole();
                            System.out.println(x:"Details: ");
                            System.out.println(media);
                            mediaDetailsMenu(scanner, media);
                            foundDetails = true;
                        } else {
                            System.out.println(x:"***MEDIA NOT FOUND***");
                        }
                    }
                    break;
                case 2:
                    boolean foundToAdd = false;
                    while (!foundToAdd) {
```

```
    public static void storeMenu(Scanner scanner) {
                case 2:
                    boolean foundToAdd = false;
                    while (!foundToAdd) {
                        System.out.println(x:"Enter the title of the media (type 0 to stop): ");
                        String title = scanner.nextLine();
                        if (title.equals(anObject:"0")) {
                            clearConsole();
                            break;
                        }
                        Media media = store.search(title);
                        if (media != null) {
                            cart.addMedia(media);
                            foundToAdd = true;
                        } else {
                            System.out.println(x:"***MEDIA NOT FOUND***");
                        }
                    }
                    break;
                case 3:
                    boolean foundToPlay = false;
                    while (!foundToPlay) {
                        System.out.println(x:"Enter the title of the media (type 0 to stop): ");
                        String title = scanner.nextLine();
                        if (title.equals(anObject:"0")) {
                            clearConsole();
                            break;
                        }
                        Media media = store.search(title);
                        if (media != null) {
                            if (media instanceof Disc || media instanceof CompactDisc) {
                                media.play();
                            } else {
                                System.out.println(x:"This type of media is not supported!");
                            }
                            foundToPlay = true;
                        } else {
                            System.out.println(x:"***MEDIA NOT FOUND***");
                        }
                    }
                    break;
                case 4:
                    clearConsole();
                    cartMenu(scanner);
                    break;
                default:
                    clearConsole();
                    System.out.println(x:"Invalid option, please choose again.");
                    break;
            }
        }
    }
```

```java
public static void mediaDetailsMenu(Scanner scanner, Media media) {
    boolean back = false;
    while (!back) {
        System.out.println(x:"Options: ");
        System.out.println(x:"--------------------------------");
        System.out.println(x:"1. Add to cart");
        System.out.println(x:"2. Play");
        System.out.println(x:"0. Back");
        System.out.println(x:"--------------------------------");
        System.out.println(x:"Please choose a number: 0-1-2");
        int option = scanner.nextInt();
        scanner.nextLine();
        switch (option) {
            case 0:
                clearConsole();
                back = true;
                break;
            case 1:
                cart.addMedia(media);
                break;
            case 2:
                if (media instanceof Disc || media instanceof CompactDisc) {
                    media.play();
                } else {
                    System.out.println(x:"This type of media is not supported!");
                }
                break;
            default:
                clearConsole();
                System.out.println(x:"Invalid option, please choose again.");
                break;
        }
    }
}
public static void cartMenu(Scanner scanner) {
    boolean back = false;
    while (!back) {
        cart.print();
        System.out.println(x:"Options: ");
        System.out.println(x:"--------------------------------");
        System.out.println(x:"1. Filter medias in cart");
        System.out.println(x:"2. Sort medias in cart");
        System.out.println(x:"3. Remove media from cart");
        System.out.println(x:"4. Play a media");
        System.out.println(x:"5. Place order");
        System.out.println(x:"0. Back");
        System.out.println(x:"--------------------------------");
        System.out.println(x:"Please choose a number: 0-1-2-3-4-5");
        int option = scanner.nextInt();
        scanner.nextLine();
        switch (option) {
            case 0:
                clearConsole();
                back = true;
```

```
                    break;
                case 1:
                    System.out.println(x:"Filter medias in cart by (1) id or (2) title:");
                    int filterOption = scanner.nextInt();
                    scanner.nextLine();
                    boolean found = false;
                    while (!found) {
                        if (filterOption == 1) {
                            System.out.println(x:"Enter the id to filter (type 0 to stop):");
                            int id = scanner.nextInt();
                            scanner.nextLine();
                            if (id == 0) {
                                clearConsole();
                                break;
                            }
                            cart.searchById(id);
                            found = true;
                        } else if (filterOption == 2) {
                            System.out.println(x:"Enter the title to filter (type 0 to stop):");
                            String title = scanner.nextLine();
                            if (title.equals(anObject:"0")) {
                                clearConsole();
                                break;
                            }
                            cart.searchByTitle(title);
                            found = true;
                        } else if (filterOption == 0) {
                            clearConsole();
                            break;
                        } else {
                            System.out.println(x:"Invalid option.");
                        }
                    }
                    break;
                case 2:
                    System.out.println(x:"Sort medias in cart by (1) title or (2) cost:");
                    int sortOption = scanner.nextInt();
                    scanner.nextLine();
                    if (sortOption == 1) {
                        cart.sortMediaByTitle();
                    } else if (sortOption == 2) {
                        cart.sortMediaByCost();
                    } else {
                        System.out.println(x:"Invalid option.");
                    }
                    break;
                case 3:
                    boolean foundToRemove = false;
                    while (!foundToRemove) {
                        System.out.println(x:"Enter the title of the media (type 0 to stop): ");
                        String title = scanner.nextLine();
                        if (title.equals(anObject:"0")) {
                            clearConsole();
```

```
                    if (title.equals(anObject:"0")) {
                        clearConsole();
                        break;
                    }
                    Media media = cart.searchToRemove(title);
                    if (media != null) {
                        clearConsole();
                        cart.removeMedia(media);
                        foundToRemove = true;
                    } else {
                        System.out.println(x:"***MEDIA NOT FOUND***");
                    }
                }
                break;
            case 4:
                boolean foundToPlay = false;
                while (!foundToPlay) {
                    System.out.println(x:"Enter the title of the media (type 0 to stop): ");
                    String title = scanner.nextLine();
                    if (title.equals(anObject:"0")) {
                        clearConsole();
                        break;
                    }
                    Media media = store.search(title);
                    if (media != null) {
                        if (media instanceof Disc || media instanceof CompactDisc) {
                            media.play();
                        } else {
                            System.out.println(x:"This type of media is not supported!");
                        }
                        foundToPlay = true;
                    } else {
                        System.out.println(x:"***MEDIA NOT FOUND***");
                    }
                }
                break;
            case 5:
                clearConsole();
                cart.empty();
                break;
            default:
                clearConsole();
                System.out.println(x:"Invalid option, please choose again.");
                break;
            }
        }
    }
    public static void updateStoreMenu(Scanner scanner) {
        boolean back = false;
        while (!back) {
            System.out.println(x:"Options: ");
            System.out.println(x:"--------------------------------");
            System.out.println(x:"1. Add a media");
            System.out.println(x:"2. Remove a media");
            System.out.println(x:"0. Back");
```

```
366             System.out.println(x:"Please choose a number: 0-1-2");
367             int option = scanner.nextInt();
368             scanner.nextLine();
369             switch (option) {
370                 case 0:
371                     clearConsole();
372                     back = true;
373                     break;
374                 case 1:
375                     System.out.println(x:"Enter the category of the media (1) Book, (2) CD, (3) DVD or (0) exit:");
376                     int categoryChoice = scanner.nextInt();
377                     scanner.nextLine();
378 
379                     if (categoryChoice == 1) {
380                         System.out.println(x:"Enter book title: ");
381                         String bookTitle = scanner.nextLine();
382                         System.out.println(x:"Enter book category: ");
383                         String bookCategory = scanner.nextLine();
384                         System.out.println(x:"Enter book cost: ");
385                         Float bookCost = scanner.nextFloat();
386                         scanner.nextLine();
387 
388                         Book newBook = new Book(bookTitle, bookCategory, bookCost);
389                         store.addMedia(newBook);
390                     } else if (categoryChoice == 2) {
391                         System.out.println(x:"Enter title: ");
392                         String cdTitle = scanner.nextLine();
393                         System.out.println(x:"Enter category: ");
394                         String cdCategory = scanner.nextLine();
395                         System.out.println(x:"Enter artist: ");
396                         String cdArtist = scanner.nextLine();
397                         System.out.println(x:"Enter cost: ");
398                         Float cdCost = scanner.nextFloat();
399                         scanner.nextLine();
400 
401                         CompactDisc newCD = new CompactDisc(cdTitle, cdCategory, cdArtist, cdCost);
402 
403 
404                         System.out.println(x:"Do you want to add tracks to your CD? (1) Yes (0) No:");
405                         int addTrack = scanner.nextInt();
406                         scanner.nextLine();
407 
408                         if (addTrack == 1) {
409                             System.out.println(x:"How many tracks in your CD?");
410                             int numTrack = scanner.nextInt();
411                             scanner.nextLine();
412                             for (int i = 0; i < numTrack; i++) {
413                                 System.out.println("Your " + (i+1) + " track: ");
414                                 System.out.println(x:"Enter track title: ");
415                                 String trackTitle = scanner.nextLine();
416                                 System.out.println(x:"Enter track length: ");
417                                 int trackLength = scanner.nextInt();
418                                 scanner.nextLine();
419 
420                                 Track newTrack = new Track(trackTitle, trackLength);
421                                 newCD.addTrack(newTrack);
```

```
                        Track newTrack = new Track(trackTitle, trackLength);
                        newCD.addTrack(newTrack);
                    }
                    store.addMedia(newCD);
                } else if (addTrack == 0) {
                    store.addMedia(newCD);
                }
            } else if (categoryChoice == 3) {
                System.out.println(x:"Enter DVD title: ");
                String dvdTitle = scanner.nextLine();
                System.out.println(x:"Enter DVD category: ");
                String dvdCategory = scanner.nextLine();
                System.out.println(x:"Enter book cost: ");
                Float dvdCost = scanner.nextFloat();
                scanner.nextLine();

                DigitalVideoDisc newDVD = new DigitalVideoDisc(dvdTitle, dvdCategory, dvdCost);
                store.addMedia(newDVD);
            } else if (categoryChoice == 0) {
                clearConsole();
                break;
            } else {
                System.out.println(x:"Invalid option.");
            }
            break;
        case 2:
            boolean foundToRemove = false;
            while (!foundToRemove) {
                System.out.println(x:"Enter the title of the media (type 0 to stop): ");
                String titleForRemove = scanner.nextLine();
                if (titleForRemove.equals(anObject:"0")) {
                    clearConsole();
                    break;
                }
                Media media = store.search(titleForRemove);
                if (media != null) {
                    clearConsole();
                    store.removeMedia(media);
                    foundToRemove = true;
                } else {
                    System.out.println(x:"***MEDIA NOT FOUND***");
                }
            }
            break;
        default:
            clearConsole();
            System.out.println(x:"Invalid option, please choose again.");
            break;
    }
  }
 }
}
```

3. Creating the **Book** class

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

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

public class Book extends Media {

    private List<String> authors = new ArrayList<String>();

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

    //getter and settter
    public List<String> getAuthors() {
        return authors;
    }
    public void setAuthors(List<String> authors) {
        this.authors = authors;
    }

    // Add and remove author
    public void addAuthor(String authorName) {
        if (!authors.contains(authorName)) {
            authors.add(authorName);
        } else {
            System.out.println("This author has already been in the list!");
        }
    }

    public void removeAuthor(String authorName) {
        if (authors.contains(authorName)) {
            authors.remove(authorName);
        } else {
            System.out.println("Not correct.Pls try again");
        }
    }

    @Override
    public String toString() {
        return this.getId() +"Book: " + this.getTitle() + " - " + this.getCategory() + " - "  + this.getCost() + " $";
    }
}
```

4. Creating the abstract **Media** class

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

import java.util.Comparator;

public abstract class Media implements Comparable<Media> {

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


    private static int nbMedia = 0;
    private int id;
    private String title;
    private String category;
    private float cost;

    // Constructor
    public Media(String title) {
        this.title = title;
        this.id = ++nbMedia;
    }
    public Media(String title, String category) {
        this.title = title;
        this.category = category;
        this.id = ++nbMedia;
    }
    public Media(String title, String category, float cost) {
        this.title = title;
        this.category = category;
        this.cost = cost;
        this.id = ++nbMedia;
    }

    // Getter method
    public int getId() {
        return id;
    }
    public String getTitle() {
        return title;
    }
    public String getCategory() {
        return category;
    }
    public float getCost() {
        return cost;
    }

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

    // Check is title match
    public boolean isMatch(String title) {
        return this.getTitle().toLowerCase().contains(title.toLowerCase());
    }

```

```
57
58      public void play() {
59          System.out.println(x:"Playing media");
60      }
61
62      @Override
63      public boolean equals(Object obj) {
64          if (obj == this) {
65              return true;
66          }
67          if (!(obj instanceof Media)) {
68              return false;
69          }
70          return ((Media)obj).getTitle() == this.getTitle();
71      }
72
73      @Override
74      public String toString() {
75          return "Media: " + this.getTitle() +
76                  " - Category: " + this.getCategory() +
77                  " - Cost: " + this.getCost() + "$";
78      }
79  }
```

5. Creating the **CompactDisc** class

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

public class CompactDisc extends Disc implements Playable {

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

    public String getArtist() {
        return artist;
    }

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

    // Add and remove track method
    public void addTrack(Track track) {
        if (!tracks.contains(track)) {
            tracks.add(track);
            System.out.println("Track: " + track.getTitle() + " has been added to CD!" );
        } else {
            System.out.println(x:"Track already exists in CD.");
        }
    }

    public void removeTrack(Track track) {
        if (tracks.contains(track)) {
            tracks.remove(track);
            System.out.println("Track: " +track.getTitle() + " has been removed from CD!");
        } else {
            System.out.println(x:"Track does not exist in CD.");
        }
    }

    // Get length method
    public int getLength() {
        int totalLength = 0;
        for (Track track : tracks) {
            totalLength += track.getLength();
        }
        return totalLength;
    }

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

```java
    }

    @Override
    public String toString() {
        return this.getId() + " - CD: " + this.getTitle() +"-" + this.getCategory() + "-"+
                this.getArtist() +"-"+ this.getLength() + "-"
                + this.getCost() + "$";
    }
}
```

6. Create the **Playable** interface

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

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

12. Sort media in the **cart**

```java
public void sortMediaByTitle() {
    Collections.sort(itemsOrdered, Media.COMPARE_BY_TITLE_COST);
    for (Media media : itemsOrdered) {
        System.out.println(media.toString());
    }
}

public void sortMediaByCost() {
    Collections.sort(itemsOrdered, Media.COMPARE_BY_COST_TITLE);
    for (Media media : itemsOrdered) {
        System.out.println(media.toString());
    }
}


//Search to remove method
public Media searchToRemove(String title) {
    for (Media media : itemsOrdered) {
        if (media.getTitle().equals(title)) {
            return media;
        }
    }
    return null;
}
```

1. What class should implement the Comparable interface?

- The media class should implement it since this is the base.

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

want?

```
@Override
public int compareTo(Media other) {
    int titleComparison = this.getTitle().compareTo(other.getTitle());
    if (titleComparison != 0) {
        return titleComparison;
    }
    return Double.compare(this.getCost(), other.getCost());
}
```

3. 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?

- No, the Comparable interface allows for only one natural ordering.

4. 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?

```
    @Override
    public int compareTo(Media other) {
        if (other instanceof Disc) {
            Disc otherDVD = (Disc) other;
```

```
        int titleComparison = this.getTitle().compareTo(otherDVD.getTitle());
        if (titleComparison != 0) {
            return titleComparison;
        } else {
            // Compare by decreasing length
            int lengthComparison = Integer.compare(otherDVD.getLength(),
this.getLength());
            if (lengthComparison != 0) {
                return lengthComparison;
            } else {
                return Double.compare(this.getCost(), otherDVD.getCost());
            }
        }
    } else {
        return super.compareTo(other);
    }
}
```