

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

**BÁO CÁO THỰC HÀNH**  
**IT3103-744530-2024.1**  
**BÀI THỰC HÀNH OOP LAB**

Họ và tên sv: Nguyễn Thanh Hưng

MSSV: 20225633

Lớp: Việt Nhật 03 K67

GVHD: Lê Thị Hoa

HTGD: Bùi Trọng Dũng

Hà Nội 11/2024

## Table of Contents

1. Create the book class .....	5
2. Creating the abstract Media class .....	7
3. Creating the CompactDisc class .....	9
3.1. Create the Disc class extending the Media clas .....	9
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.....	11
3.3. Open the CompactDisc class .....	12
4. Create the Playable interface .....	13
5. Update the Cart class to work with Media .....	14
6. Update the Store class to work with Media .....	19
7. Constructors of whole classes and parent classes.....	21
8. Unique item in a list.....	21
9. Polymorphism with toString() method .....	22
10. Sort media in the cart.....	23
11. Create a complete console application in the Aims class .....	24
11.1. Người dùng chọn 1: View store .....	24
11.2. Người dùng chọn 2: Update store .....	27
11.3. Người dùng chọn 3: See current cart .....	29
12. Class diagram .....	33
13. UseCase diagram .....	34
14. Answer Question .....	35

## Table of Figures

<u>Figure 1: Book Class</u> .....	7
<u>Figure 2.1: Media Class</u> .....	9
<u>Figure 2.2: Media Class</u> .....	9
<u>Figure 3: Disc Class</u> .....	10
<u>Figure 4: DigitalVideoDisc Class</u> .....	11
<u>Figure 5: Track Class</u> .....	12
<u>Figure 6.1: CompactDisc Class</u> .....	13
<u>Figure 6.2: CompactDisc Class</u> .....	13
<u>Figure 7: Playable interface</u> .....	13
<u>Figure 8: Method play() của CompactDisc</u> .....	14
<u>Figure 9: Method play() của DigitalVideoDisc</u> .....	14
<u>Figure 10: Method play() của Track</u> .....	14
<u>Figure 11.1: Cart Class</u> .....	15
<u>Figure 11.2: Cart Class</u> .....	17
<u>Figure 11.3: Cart Class</u> .....	18
<u>Figure 11.4: Cart Class</u> .....	19
<u>Figure 11.5: Cart Class</u> .....	19
<u>Figure 12.1: Store Class</u> .....	20
<u>Figure 12.2: Store Class</u> .....	21
<u>Figure 13: Constructors Track Class</u> .....	21
<u>Figure 14: Constructors CompactDisc Class</u> .....	21
<u>Figure 15: Constructors Media Class</u> .....	21
<u>Figure 16: Constructors Disc Class</u> .....	21
<u>Figure 17: Override equals in Media Class</u> .....	22
<u>Figure 18: Override equals in Track Class</u> .....	22
<u>Figure 19: Code mô phỏng Polymorphism</u> .....	23
<u>Figure 20: Override toString() in Media Class</u> .....	23
<u>Figure 21: Result</u> .....	23
<u>Figure 22: Add the comparators as attributes of the Media class</u> .....	24
<u>Figure 23: Media ComparatorByTitleCost Class</u> .....	24
<u>Figure 24: Media ComparatorByCostTitle Class</u> .....	24
<u>Figure 25: Màn hình chính</u> .....	24

<u>Figure 26: Vào trang View store</u> .....	25
<u>Figure 27: See a media's details</u> .....	25
<u>Figure 28: Thêm vào Cart</u> .....	26
<u>Figure 29: Thêm media vào Cart</u> .....	26
<u>Figure 29: Play a media</u> .....	26
<u>Figure 30: See current cart after filter</u> .....	27
<u>Figure 31: Add a media to store</u> .....	28
<u>Figure 32: Kết quả sau khi thêm</u> .....	28
<u>Figure 33: Remove a media from the store and result</u> .....	29
<u>Figure 34: Trong cart hiện tại</u> .....	30
<u>Figure 35: Filter cart by Title</u> .....	30
<u>Figure 36: Sort cart by Title</u> .....	31
<u>Figure 37: Remove media from cart</u> .....	32
<u>Figure 38: Play a media</u> .....	33
<u>Figure 39: Place order</u> .....	33
<u>Figure 40: Class diagram</u> .....	34
<u>Figure 41: UseCase diagram</u> .....	35

# BÁO CÁO THỰC HÀNH LAB 04

## LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

### 1. Create the book class

```

Media_NTH.java Disc_NTH.java CompactDisc_NTH.java Track_NTH.java Playable.java
1 package hust.soict.hedspi.aims.media;
2
3 import java.util.ArrayList;
4
5 public class Book_NTH extends Media_NTH {
6     private ArrayList<String> authors;
7
8     // Constructor
9     public Book_NTH(int id, String title, String category, float cost) {
10         super(id, title, category, cost); // Gọi constructor của lớp Media_NTH
11         this.authors = new ArrayList<>();
12     }
13
14     // Getter và Setter cho authors
15     public ArrayList<String> getAuthors() {
16         return authors;
17     }
18
19     public void setAuthors(ArrayList<String> authors) {
20         if (authors != null) {
21             this.authors = authors;
22         } else {
23             System.out.println("Cannot set null to authors list.");
24         }
25     }
26
27     // Phương thức addAuthor và removeAuthor
28     public void addAuthor(String authorName) {
29         if (authorName == null || authorName.trim().isEmpty()) {
30             System.out.println("Author name cannot be empty.");
31         } else if (!authors.contains(authorName)) {
32             authors.add(authorName);
33             System.out.println("Author " + authorName + " added to the book.");
34         } else {
35             System.out.println("Author " + authorName + " is already in the list.");
36         }
37     }
38
39     public void removeAuthor(String authorName) {
40         if (authorName == null || authorName.trim().isEmpty()) {
41             System.out.println("Author name cannot be empty.");
42         } else if (authors.contains(authorName)) {
43             authors.remove(authorName);
44             System.out.println("Author " + authorName + " removed from the book.");
45         } else {
46             System.out.println("Author " + authorName + " not found in the list.");
47         }
48     }
49
50     // Override phương thức displayInfo
51     @Override
52     public void displayInfo() {
53         System.out.println("Book ID: " + getId());
54         System.out.println("Title: " + getTitle());
55         System.out.println("Category: " + getCategory());
56         System.out.println("Cost: " + getCost());
57         System.out.println("Authors: " + authors);
58     }
59 }

```

## Figure 1: Book Class

### 2. Creating the abstract Media class

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

```
Media_NTH.java × Disc_NTH.java CompactDisc_NTH.java Track_NTH.java Playable.java
1 package hust.soict.hedspi.aims.media;
2
3 import java.util.Objects;
4 import java.util.Comparator;
5
6 public abstract class Media_NTH {
7     private int id;
8     private String title;
9     private String category;
10    private float cost;
11
12    // Constructor
13    public Media_NTH(int id, String title, String category, float cost) {
14        this.id = id;
15        this.title = title;
16        this.category = category;
17        this.cost = cost;
18    }
19
20    // Getter và Setter
21    public int getId() {
22        return id;
23    }
24
25    public void setId(int id) {
26        this.id = id;
27    }
28
29    public String getTitle() {
30        return title;
31    }
32
33    public void setTitle(String title) {
34        this.title = title;
35    }
36
37    public String getCategory() {
38        return category;
39    }
40
41    public void setCategory(String category) {
42        this.category = category;
43    }
44
45    public float getCost() {
46        return cost;
47    }
48
49    public void setCost(float cost) {
50        this.cost = cost;
51    }
52
53    // Phương thức abstract để mỗi loại media sẽ có phương thức hiển thị riêng
54    public abstract void displayInfo();
55
56    public boolean isMatch(String title) {
57        return this.title.equalsIgnoreCase(title); // So sánh tên media với tiêu đề
58    }
```



Figure 2.1: Media Class

```

59
60 // Ghi đè phương thức equals() để so sánh các đối tượng Media_NTH
61 @Override
62 public boolean equals(Object obj) {
63     if (this == obj) return true;
64     if (obj == null || getClass() != obj.getClass()) return false;
65
66     Media_NTH media = (Media_NTH) obj;
67     return Objects.equals(title, media.title); // So sánh theo title
68 }
69
70 @Override
71 public int hashCode() {
72     return Objects.hash(title); // Tạo hashCode dựa trên title
73 }
74
75 // Phương thức toString() để in thông tin cơ bản
76 @Override
77 public String toString() {
78     return "Title: " + title + " | Category: " + category + " | Cost: $" + cost;
79 }
80
81 public static final Comparator<Media_NTH> COMPARE_BY_TITLE_COST =
82     Comparator.comparing(Media_NTH::getTitle)
83         .thenComparing(Comparator.comparing(Media_NTH::getCost).reversed());
84
85 public static final Comparator<Media_NTH> COMPARE_BY_COST_TITLE =
86     Comparator.comparing(Media_NTH::getCost).reversed()
87         .thenComparing(Media_NTH::getTitle);
88
89
90 public int compareTo(Media_NTH other) {
91     // So sánh Tiêu đề
92     int titleCompare = this.title.compareTo(other.title);
93
94     // Nếu Tiêu đề khác nhau, trả về kết quả so sánh Tiêu đề
95     if (titleCompare != 0) {
96         return titleCompare;
97     }
98
99     // Nếu Tiêu đề giống nhau, sắp xếp theo Chi phí (Giảm dần)
100     return Float.compare(other.cost, this.cost); // Chi phí giảm dần
101 }
102 }

```

Figure 2.2: Media Class

### 3. Creating the CompactDisc class

#### 3.1. Create the Disc class extending the Media clas

```

Media_NTH.java Disc_NTH.java × CompactDisc_NTH.java Track_NTH.java Playable.java Cart_NTH.j
1 package hust.soict.hedspi.aims.media;
2
3 public abstract class Disc_NTH extends Media_NTH {
4     private int length;
5     private String director;
6
7     // Constructor
8     public Disc_NTH(int id, String title, String category, float cost, int length, String director) {
9         super(id, title, category, cost);
10        this.length = length;
11        this.director = director;
12    }
13
14    // Getter và Setter
15    public int getLength() {
16        return length;
17    }
18
19    public void setLength(int length) {
20        this.length = length;
21    }
22
23    public String getDirector() {
24        return director;
25    }
26
27    public void setDirector(String director) {
28        this.director = director;
29    }
30
31    // Phương thức displayInfo() để hiển thị thông tin cơ bản
32    @Override
33    public void displayInfo() {
34        System.out.println("Title: " + getTitle());
35        System.out.println("Category: " + getCategory());
36        System.out.println("Cost: $" + getCost());
37        System.out.println("Length: " + length + " minutes");
38        System.out.println("Director: " + director);
39    }
40 }
41

```

Figure 3: Disc Class

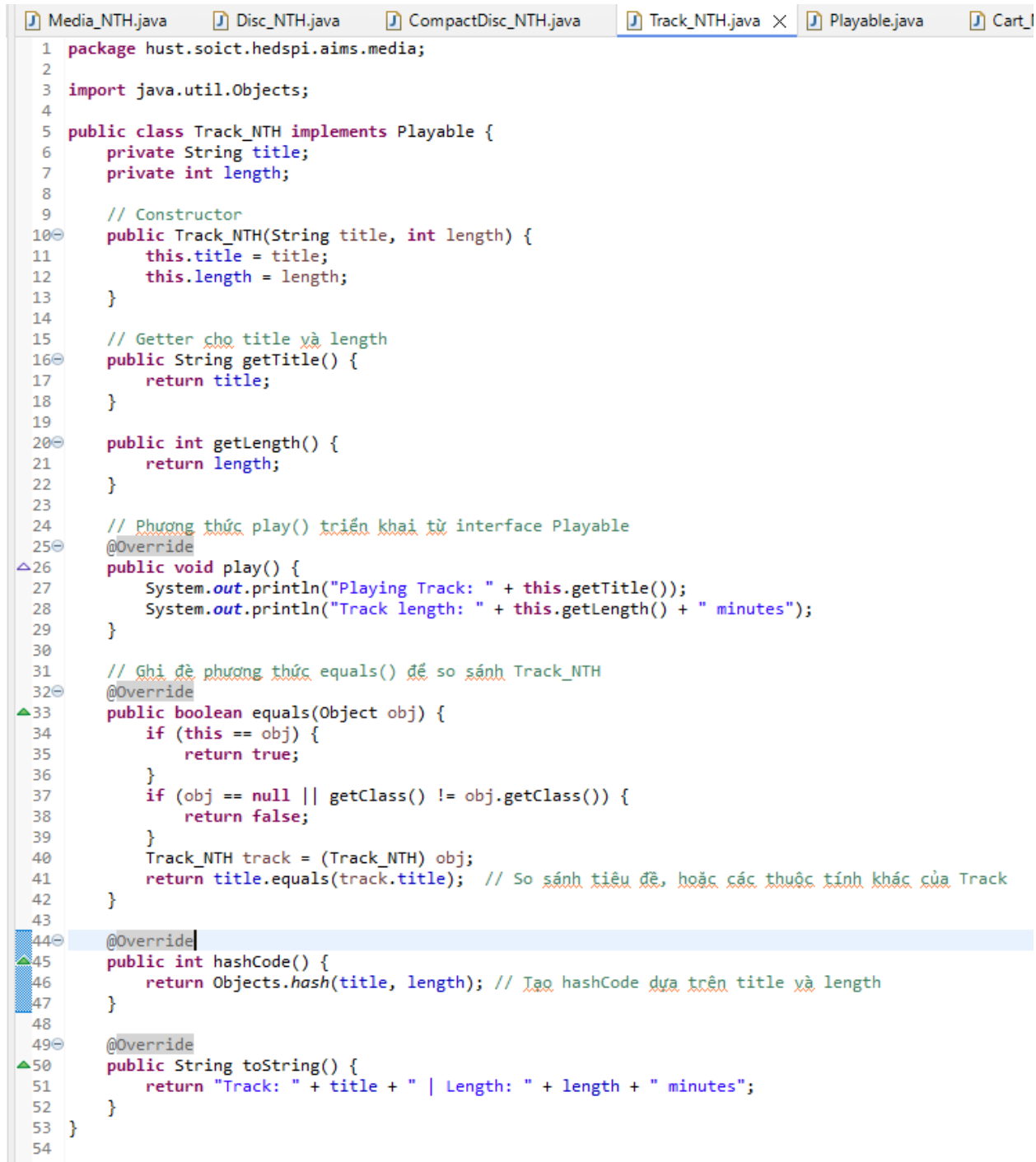
```

Media_NTH.java × Disc_NTH.java CompactDisc_NTH.java Track_NTH.java Playable.java Cart_NTH.java Ai
1 import hust.soict.hedspi.aims.media.Disc_NTH;
2 import hust.soict.hedspi.aims.media.Playable;
3
4 public class DigitalVideoDisc_NTH extends Disc_NTH implements Playable {
5
6     // Constructor
7     public DigitalVideoDisc_NTH(int id, String title, String category, String director, int length, float cost) {
8         super(id, title, category, cost, length, director); // Gọi constructor của lớp Disc_NTH
9     }
10
11    // Phương thức play() của interface Playable
12    @Override
13    public void play() {
14        System.out.println("Playing DVD: " + this.getTitle());
15        System.out.println("DVD length: " + this.getLength() + " minutes");
16    }
17
18    // Phương thức displayInfo() để hiển thị thông tin của DVD
19    @Override
20    public void displayInfo() {
21        super.displayInfo();
22    }
23 }
24

```

Figure 4: DigitalVideoDisc Class

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



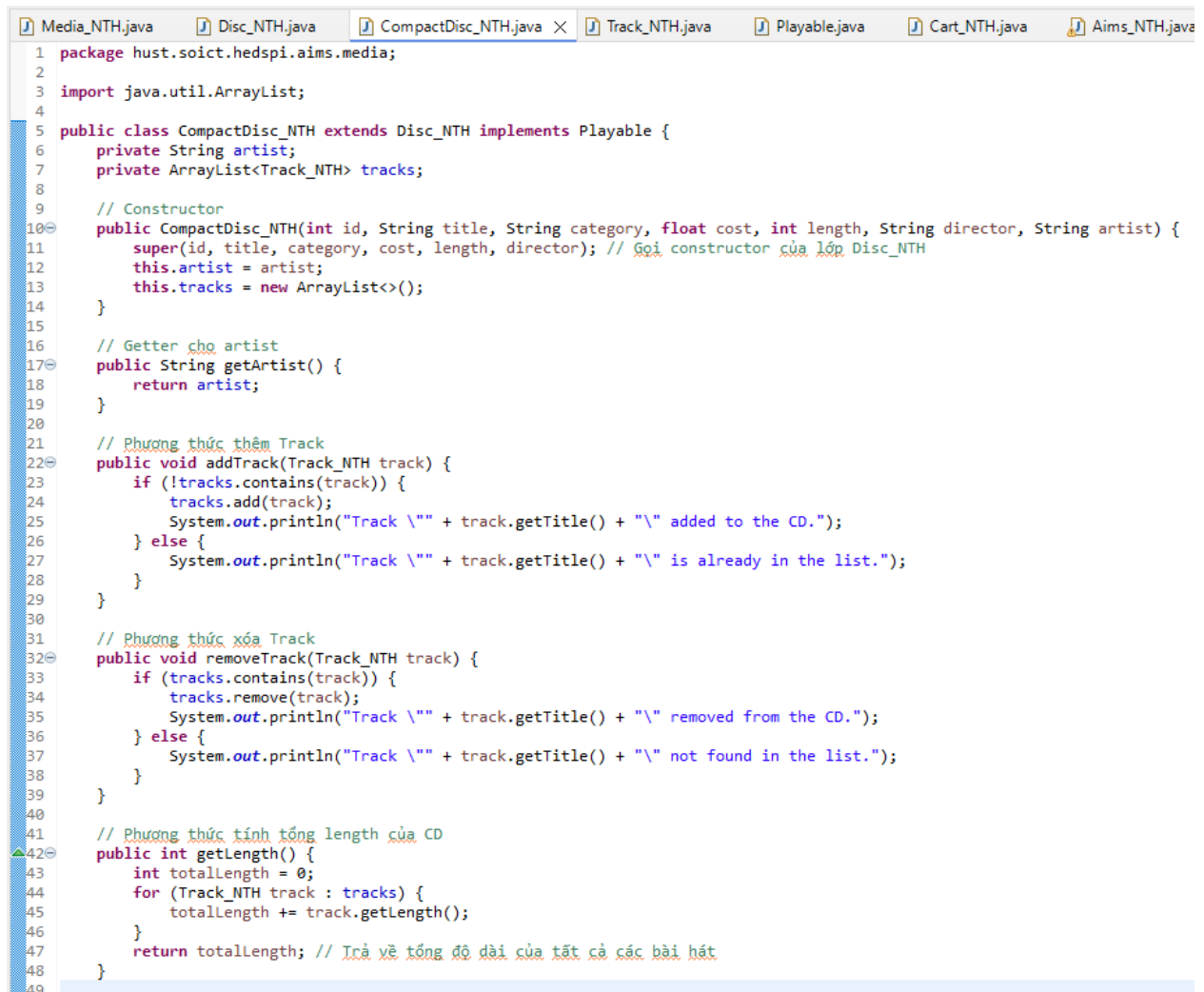
```

1 package hust.soict.hedspi.aims.media;
2
3 import java.util.Objects;
4
5 public class Track_NTH implements Playable {
6     private String title;
7     private int length;
8
9     // Constructor
10    public Track_NTH(String title, int length) {
11        this.title = title;
12        this.length = length;
13    }
14
15    // Getter cho title và length
16    public String getTitle() {
17        return title;
18    }
19
20    public int getLength() {
21        return length;
22    }
23
24    // Phương thức play() triển khai từ interface Playable
25    @Override
26    public void play() {
27        System.out.println("Playing Track: " + this.getTitle());
28        System.out.println("Track length: " + this.getLength() + " minutes");
29    }
30
31    // Ghi đè phương thức equals() để so sánh Track_NTH
32    @Override
33    public boolean equals(Object obj) {
34        if (this == obj) {
35            return true;
36        }
37        if (obj == null || getClass() != obj.getClass()) {
38            return false;
39        }
40        Track_NTH track = (Track_NTH) obj;
41        return title.equals(track.title); // So sánh tiêu đề, hoặc các thuộc tính khác của Track
42    }
43
44    @Override
45    public int hashCode() {
46        return Objects.hash(title, length); // Tạo hashCode dựa trên title và length
47    }
48
49    @Override
50    public String toString() {
51        return "Track: " + title + " | Length: " + length + " minutes";
52    }
53 }
54

```

Figure 5: Track Class

### 3.3. Open the CompactDisc class



```

1 package hust.soict.hedspi.aims.media;
2
3 import java.util.ArrayList;
4
5 public class CompactDisc_NTH extends Disc_NTH implements Playable {
6     private String artist;
7     private ArrayList<Track_NTH> tracks;
8
9     // Constructor
10    public CompactDisc_NTH(int id, String title, String category, float cost, int length, String director, String artist) {
11        super(id, title, category, cost, length, director); // Gọi constructor của lớp Disc_NTH
12        this.artist = artist;
13        this.tracks = new ArrayList<>();
14    }
15
16    // Getter cho artist
17    public String getArtist() {
18        return artist;
19    }
20
21    // Phương thức thêm Track
22    public void addTrack(Track_NTH track) {
23        if (!tracks.contains(track)) {
24            tracks.add(track);
25            System.out.println("Track \"" + track.getTitle() + "\" added to the CD.");
26        } else {
27            System.out.println("Track \"" + track.getTitle() + "\" is already in the list.");
28        }
29    }
30
31    // Phương thức xóa Track
32    public void removeTrack(Track_NTH track) {
33        if (tracks.contains(track)) {
34            tracks.remove(track);
35            System.out.println("Track \"" + track.getTitle() + "\" removed from the CD.");
36        } else {
37            System.out.println("Track \"" + track.getTitle() + "\" not found in the list.");
38        }
39    }
40
41    // Phương thức tính tổng length của CD
42    public int getLength() {
43        int totalLength = 0;
44        for (Track_NTH track : tracks) {
45            totalLength += track.getLength();
46        }
47        return totalLength; // Trả về tổng độ dài của tất cả các bài hát
48    }
49

```

Figure 6.1: CompactDisc Class

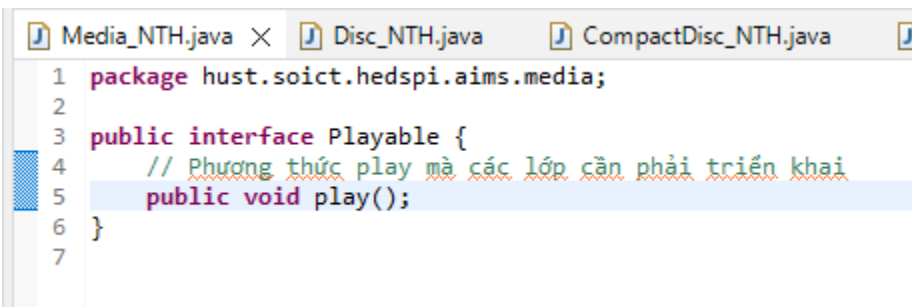
```

49
50 // Phương thức play()
51 @Override
52 public void play() {
53     System.out.println("Playing CD: " + this.getTitle());
54     System.out.println("Artist: " + artist);
55     System.out.println("Total Length: " + getLength() + " minutes");
56
57     // Phát từng track trong danh sách
58     for (Track_NTH track : tracks) {
59         track.play(); // Gọi phương thức play() của Track
60     }
61 }
62
63 // Phương thức displayInfo()
64 @Override
65 public void displayInfo() {
66     super.displayInfo();
67     System.out.println("Artist: " + artist);
68     System.out.println("Total Length: " + getLength() + " minutes");
69     System.out.println("Tracks: ");
70     for (Track_NTH track : tracks) {
71         System.out.println(" - " + track.getTitle() + " (" + track.getLength() + " minutes)");
72     }
73 }
74 }

```

Figure 6.2: CompactDisc Class

#### 4. Create the Playable interface



```

Media_NTH.java × Disc_NTH.java CompactDisc_NTH.java
1 package hust.soict.hedspi.aims.media;
2
3 public interface Playable {
4     // Phương thức play mà các lớp cần phải triển khai
5     public void play();
6 }
7

```

Figure 7: Playable interface

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

```

50 // Phương thức play()
51 @Override
52 public void play() {
53     System.out.println("Playing CD: " + this.getTitle());
54     System.out.println("Artist: " + artist);
55     System.out.println("Total Length: " + getLength() + " minutes");
56
57     // Phát từng track trong danh sách
58     for (Track_NTH track : tracks) {
59         track.play(); // Gọi phương thức play() của Track
60     }
61 }

```

Figure 8: Method play() của CompactDisc

```
11 // Phương thức play() của interface Playable
12 @Override
13 public void play() {
14     System.out.println("Playing DVD: " + this.getTitle());
15     System.out.println("DVD length: " + this.getLength() + " minutes");
16 }
```

Figure 9: Method play() của DigitalVideoDisc

```
25 @Override
26 public void play() {
27     System.out.println("Playing Track: " + this.getTitle());
28     System.out.println("Track length: " + this.getLength() + " minutes");
29 }
30
```

Figure 10: Method play() của Track

## 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.



```

1  import hust.soict.hedspi.aims.media.*;
2  import java.util.ArrayList;
3  import java.util.Random;
4  import java.util.Collections;
5
6  public class Cart_NTH {
7      private ArrayList<Media_NTH> itemsOrdered; // Giỏ hàng giờ chứa Media_NTH thay vì chỉ DVD
8      private final int MAX_NUMBERS_ORDERED = 20;
9      private String orderState = "Pending";
10     private String deliDest;
11     private float deliFee;
12     private float vat;
13
14     public Cart_NTH() {
15         itemsOrdered = new ArrayList<>();
16     }
17
18     // Getter và setter cho các thuộc tính
19     public String getOrderState() {
20         return orderState;
21     }
22
23     public void setOrderState(String orderState) {
24         this.orderState = orderState;
25     }
26
27     public String getDeliDest() {
28         return deliDest;
29     }
30
31     public void setDeliDest(String deliDest) {
32         this.deliDest = deliDest;
33     }
34
35     public float getDeliFee() {
36         return deliFee;
37     }
38
39     public void setDeliFee(float deliFee) {
40         this.deliFee = deliFee;
41     }
42
43     public float getVat() {
44         return vat;
45     }
46
47     public void setVat(float vat) {
48         this.vat = vat;
49     }
50
51     // Thêm media vào giỏ hàng
52     public void addMedia(Media_NTH media) {
53         if (!itemsOrdered.contains(media)) {
54             if (itemsOrdered.size() < MAX_NUMBERS_ORDERED) {
55                 itemsOrdered.add(media);
56                 System.out.println("\nThe media \"" + media.getTitle() + "\" has been added to the cart.");
57             } else {
58                 System.out.println("\nThe cart is full. Cannot add more media.");
59             }
60         }
61     }
62 }

```

Figure 11.1: Cart Class

```

59     }
60     } else {
61         System.out.println("\nThe media \"" + media.getTitle() + "\" is already in the cart.");
62     }
63 }
64
65 // Xóa media khỏi giỏ hàng
66 public void removeMedia(Media_NTH media) {
67     if (itemsOrdered.remove(media)) {
68         System.out.println("\nThe media \"" + media.getTitle() + "\" has been removed from the cart.");
69     } else {
70         System.out.println("\nThe media is not found in the cart.");
71     }
72 }
73 public void removeMediaByTitle(String titleToRemove) {
74     Media_NTH mediaToRemove = null;
75
76     // Tìm kiếm media trong giỏ hàng theo tên
77     for (Media_NTH media : itemsOrdered) {
78         if (media.getTitle().equalsIgnoreCase(titleToRemove)) {
79             mediaToRemove = media;
80             break; // Ngừng tìm kiếm khi đã tìm thấy
81         }
82     }
83
84     // Nếu tìm thấy, gọi phương thức removeMedia với đối tượng media
85     if (mediaToRemove != null) {
86         removeMedia(mediaToRemove); // Gọi phương thức removeMedia đã có
87     } else {
88         System.out.println("No media found with title: \"" + titleToRemove + "\"");
89     }
90 }
91
92 // Tính tổng chi phí của giỏ hàng
93 public float totalCost() {
94     float total = 0;
95     for (Media_NTH media : itemsOrdered) {
96         total += media.getCost();
97     }
98     total += deliFee + total * (vat / 100); // Thêm phí giao hàng và VAT
99     return total;
100 }
101
102 // Hiển thị giỏ hàng
103 public void showCart() {
104     System.out.println("\n--- Cart Contents ---");
105     for (Media_NTH media : itemsOrdered) {
106         media.displayInfo(); // Gọi phương thức displayInfo của mỗi media
107     }
108     System.out.println("\n-----");
109     System.out.println("Total cost: $" + totalCost());
110     System.out.println("Total items in cart: " + itemsOrdered.size());
111 }
112
113 // Sắp xếp giỏ hàng theo Tiêu đề rồi đến Chi phí
114 public void sortByTitleCost() {
115     Collections.sort(itemsOrdered, Media_NTH.COMPARE_BY_TITLE_COST);
116 }

```



Figure 11.2: Cart Class

```

117
118 // Sắp xếp giỏ hàng theo Chi phí rồi đến Tiêu đề
119 public void sortByCostTitle() {
120     Collections.sort(itemsOrdered, Media_NTH.COMPARE_BY_COST_TITLE);
121 }
122
123 // Tăng ngẫu nhiên một media
124 public Media_NTH randomMedia() {
125     if (!itemsOrdered.isEmpty()) {
126         Random random = new Random();
127         int index = random.nextInt(itemsOrdered.size());
128         System.out.println("\nYou received a free media: \"" + itemsOrdered.get(index).getTitle() + "\"");
129         return itemsOrdered.get(index);
130     }
131     return null;
132 }
133
134 // Đặt hàng
135 public void placeOrder() {
136     if (!itemsOrdered.isEmpty()) {
137         System.out.println("\nOrder placed successfully with destination: " + deliDest);
138         itemsOrdered.clear(); // Giả sử đơn hàng hoàn thành và làm trống giỏ hàng
139     } else {
140         System.out.println("\nNo items in the cart to place an order.");
141     }
142 }
143
144 // Phê duyệt đơn hàng
145 public void approveOrder() {
146     if (orderState.equalsIgnoreCase("Pending")) {
147         orderState = "Approved";
148         System.out.println("\nOrder has been approved.");
149     } else {
150         System.out.println("\nOrder is not in a state that can be approved.");
151     }
152 }
153
154 // Từ chối đơn hàng
155 public void rejectOrder() {
156     if (orderState.equalsIgnoreCase("Pending")) {
157         orderState = "Rejected";
158         System.out.println("\nOrder has been rejected.");
159     } else {
160         System.out.println("\nOrder is not in a state that can be rejected.");
161     }
162 }
163
164 // Tìm kiếm media theo ID
165 public void searchById(int id) {
166     boolean found = false;
167     for (Media_NTH media : itemsOrdered) {
168         if (media.getId() == id) {
169             System.out.println("Found media with ID " + id + ":");
170             media.displayInfo();
171             found = true;
172             break;
173         }
174     }
175 }

```

Figure 11.3: Cart Class

```

173     }
174 }
175 if (!found) {
176     System.out.println("No media found with ID " + id);
177 }
178 }
179
180 // Tìm kiếm media theo tên
181 public void searchByTitle(String title) {
182     boolean found = false;
183     for (Media_NTH media : itemsOrdered) {
184         if (media.isMatch(title)) {
185             System.out.println("Found media with title \"" + title + "\"");
186             media.displayInfo();
187             found = true;
188         }
189     }
190     if (!found) {
191         System.out.println("No media found with title \"" + title + "\"");
192     }
193 }
194
195 // Lọc media theo ID
196 public void filterById(int id) {
197     System.out.println("\n--- Filter Results by ID: " + id + " ---");
198     boolean found = false;
199     for (Media_NTH media : itemsOrdered) {
200         if (media.getId() == id) {
201             media.displayInfo();
202             found = true;
203         }
204     }
205     if (!found) {
206         System.out.println("No media found with ID " + id);
207     }
208 }
209
210 // Lọc media theo Tiêu đề
211 public void filterByTitle(String title) {
212     System.out.println("\n--- Filter Results by Title: " + title + " ---");
213     boolean found = false;
214     for (Media_NTH media : itemsOrdered) {
215         if (media.isMatch(title)) {
216             media.displayInfo();
217             found = true;
218         }
219     }
220     if (!found) {
221         System.out.println("No media found with title \"" + title + "\"");
222     }
223 }
224
225 // Phát media trong giỏ hàng
226 public void playMedia(String title) {
227     boolean found = false;
228     for (Media_NTH media : itemsOrdered) {
229         if (media.isMatch(title)) {
230             if (media instanceof Playable) {

```

Figure 11.4: Cart Class

```
231         ((Playable) media).play(); // Goi phuong thuc play() của media
232         found = true;
233     } else {
234         System.out.println("This media cannot be played.");
235         found = true;
236     }
237     break;
238 }
239 }
240 if (!found) {
241     System.out.println("No media found with title \"" + title + "\"");
242 }
243 }
244 }
245 }
```

Figure 11.5: Cart Class

## 6. Update the Store class to work with Media

```

Media_NTH.java Disc_NTH.java CompactDisc_NTH.java Track_NTH.java Playable.java Cart_NTH.java Aims_NTH.java
1 import hust.soict.hedspi.aims.media.*;
2 import java.util.ArrayList;
3
4 public class Store_NTH {
5     private ArrayList<Media_NTH> itemsInStore; // Giữ chứa Media_NTH thay vì chỉ DVD
6     private final int MAX_MEDIA = 100;
7
8     public Store_NTH() {
9         itemsInStore = new ArrayList<>();
10    }
11
12    // Thêm media vào kho
13    public void addMedia(Media_NTH media) {
14        if (itemsInStore.size() < MAX_MEDIA) {
15            itemsInStore.add(media);
16            System.out.println("\nThe media \"" + media.getTitle() + "\" has been added to the store.");
17        } else {
18            System.out.println("\nStore capacity reached. Cannot add more media.");
19        }
20    }
21
22    // Xóa media khỏi kho
23    public void removeMedia(Media_NTH media) {
24        if (itemsInStore.remove(media)) {
25            System.out.println("\nThe media \"" + media.getTitle() + "\" has been removed from the store.");
26        } else {
27            System.out.println("\nThe media is not found in the store.");
28        }
29    }
30
31    // Hiển thị tất cả media trong kho
32    public void showItemsInStore() {
33        System.out.println("\n--- Media in Store ---");
34        for (Media_NTH media : itemsInStore) {
35            System.out.print "[" + media.getTitle() + " | " + media.getCategory() + " | $" + media.getCost() + " ] ";
36        }
37        System.out.println("\n-----");
38    }
39
40    // Tìm kiếm media theo tiêu đề
41    public void searchByTitle(String title) {
42        System.out.println("\n--- Search Results for Title: \"" + title + "\" ---");
43        for (Media_NTH media : itemsInStore) {
44            if (media.getTitle().equalsIgnoreCase(title)) {
45                media.displayInfo();
46                System.out.println("-----");
47            }
48        }
49    }
50
51    // Tìm kiếm media theo thể loại
52    public void searchByCategory(String category) {
53        System.out.println("\n--- Search Results for Category: \"" + category + "\" ---");
54        for (Media_NTH media : itemsInStore) {
55            if (media.getCategory().equalsIgnoreCase(category)) {
56                System.out.print "[" + media.getTitle() + " | " + media.getCategory() + " | $" + media.getCost() + " ] ";
57            }
58        }
59        System.out.println("\n-----");

```

Figure 12.1: Store Class

```

60    }
61
62    // Tìm kiếm media theo giá
63    public void searchByPrice(float price) {
64        System.out.println("\n--- Search Results for Media under: $" + price + " ---");
65        for (Media_NTH media : itemsInStore) {
66            if (media.getCost() <= price) {
67                System.out.print "[" + media.getTitle() + " | $" + media.getCost() + " ] ";
68            }
69        }
70        System.out.println("\n-----");
71    }
72 }
73

```

Figure 12.2: Store Class

## 7. Constructors of whole classes and parent classes

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

Figure 13: Constructors Track Class

```
// Constructor
public CompactDisc_NTH(int id, String title, String category, float cost, int length, String director, String artist) {
    super(id, title, category, cost, length, director); // Gọi constructor của lớp Disc_NTH
    this.artist = artist;
    this.tracks = new ArrayList<>();
}
```

Figure 14: Constructors CompactDisc Class

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

```
// Constructor
public Media_NTH(int id, String title, String category, float cost) {
    this.id = id;
    this.title = title;
    this.category = category;
    this.cost = cost;
}
```

Figure 15: Constructors Media Class

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

Figure 16: Constructors 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 lặp vào danh sách.

```

60 // Ghi đè phương thức equals() để so sánh các đối tượng Media_NTH
61 @Override
62 public boolean equals(Object obj) {
63     if (this == obj) return true;
64     if (obj == null || getClass() != obj.getClass()) return false;
65
66     Media_NTH media = (Media_NTH) obj;
67     return Objects.equals(title, media.title); // So sánh theo title
68 }
69

```

Figure 17: Override equals in Media Class

```

31 // Ghi đè phương thức equals() để so sánh Track_NTH
32 @Override
33 public boolean equals(Object obj) {
34     if (this == obj) {
35         return true;
36     }
37     if (obj == null || getClass() != obj.getClass()) {
38         return false;
39     }
40     Track_NTH track = (Track_NTH) obj;
41     return title.equals(track.title); // So sánh tiêu đề, hoặc các thuộc tính khác của Track
42 }
43

```

Figure 18: Override equals in Track Class

## 9. Polymorphism with toString() method

```

Media_NTH.java Disc_NTH.java CompactDisc_N... Track_NTH.java Playable.java Cart_NTH.java Aims_NTH.java D
1 import hust.soict.hedspi.aims.media.*;
2 import java.util.ArrayList;
3 import java.util.List;
4
5 public class Test {
6     public static void main(String[] args) {
7         // Tạo các đối tượng Media
8         DigitalVideoDisc_NTH dvd = new DigitalVideoDisc_NTH(1, "Inception", "Sci-Fi", "Christopher Nolan", 148, 19.99f);
9         Book_NTH book = new Book_NTH(2, "The Great Gatsby", "Literature", 9.99f);
10        CompactDisc_NTH cd = new CompactDisc_NTH(3, "Best of Pop", "Pop", 12.99f, 60, "John Doe", "Various Artists");
11
12        // Thêm track vào CD
13        cd.addTrack(new Track_NTH("Track 1", 3));
14        cd.addTrack(new Track_NTH("Track 2", 4));
15
16        // Tạo một danh sách Media
17        List<Media_NTH> mediaList = new ArrayList<>();
18        mediaList.add(dvd);
19        mediaList.add(book);
20        mediaList.add(cd);
21
22        // Duyệt qua danh sách và in thông tin media
23        for (Media_NTH media : mediaList) {
24            System.out.println(media.toString());
25        }
26    }
27 }
28

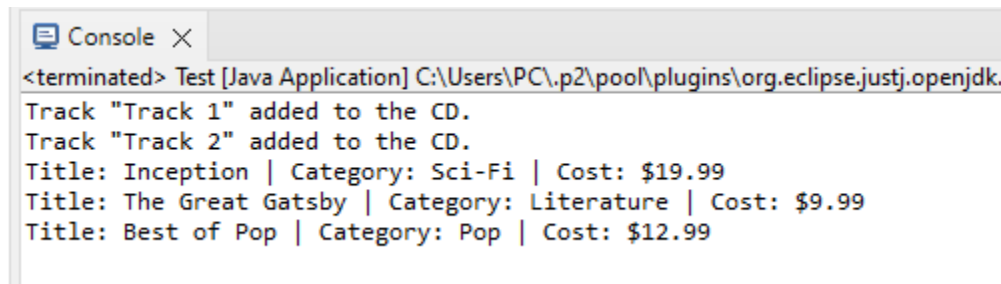
```

Figure 19: Code mô phỏng Polymorphism

```
// Phương thức toString() để in thông tin cơ bản
@Override
public String toString() {
    return "Title: " + title + " | Category: " + category + " | Cost: $" + cost;
}
```

Figure 20: Override toString() in Media Class

Kết quả



```
<terminated> Test [Java Application] C:\Users\PC\p2\pool\plugins\org.eclipse.justj.openjdk.
Track "Track 1" added to the CD.
Track "Track 2" added to the CD.
Title: Inception | Category: Sci-Fi | Cost: $19.99
Title: The Great Gatsby | Category: Literature | Cost: $9.99
Title: Best of Pop | Category: Pop | Cost: $12.99
```

Figure 21: Result

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:

- Bảng 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.
- Bảng cost: Hiển thị theo thứ tự cost giảm dần. Trong trường hợp cost như nhau, sắp xếp media theo thứ tự bảng chữ cái

```
81 public static final Comparator<Media_NTH> COMPARE_BY_TITLE_COST = new CompareByTitleCost();
82 public static final Comparator<Media_NTH> COMPARE_BY_COST_TITLE = new CompareByCostTitle();
```

Figure 22: Add the comparators as attributes of the Media class

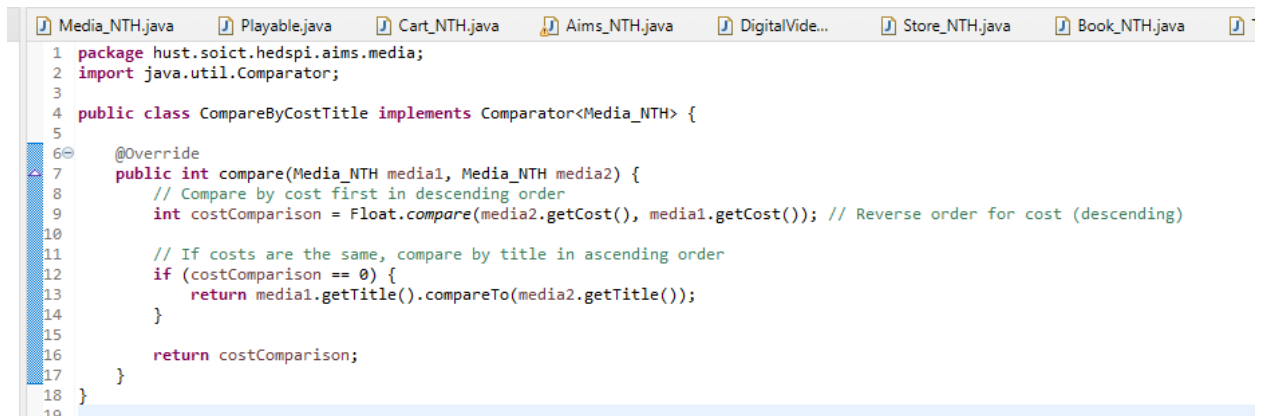


```

1 package hust.soict.hedspi.aims.media;
2 import java.util.Comparator;
3
4 public class CompareByTitleCost implements Comparator<Media_NTH> {
5
6     @Override
7     public int compare(Media_NTH media1, Media_NTH media2) {
8         // Compare by title first
9         int titleComparison = media1.getTitle().compareTo(media2.getTitle());
10
11         // If titles are the same, compare by cost in descending order
12         if (titleComparison == 0) {
13             return Float.compare(media2.getCost(), media1.getCost()); // Reverse order for cost (descending)
14         }
15
16         return titleComparison;
17     }
18 }
19

```

Figure 23: Media ComparatorByTitleCost Class



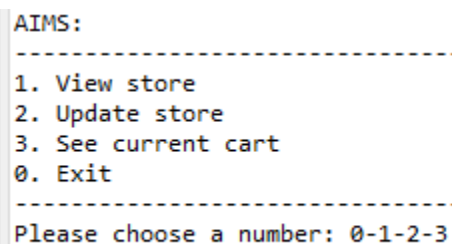
```

1 package hust.soict.hedspi.aims.media;
2 import java.util.Comparator;
3
4 public class CompareByCostTitle implements Comparator<Media_NTH> {
5
6     @Override
7     public int compare(Media_NTH media1, Media_NTH media2) {
8         // Compare by cost first in descending order
9         int costComparison = Float.compare(media2.getCost(), media1.getCost()); // Reverse order for cost (descending)
10
11         // If costs are the same, compare by title in ascending order
12         if (costComparison == 0) {
13             return media1.getTitle().compareTo(media2.getTitle());
14         }
15
16         return costComparison;
17     }
18 }
19

```

Figure 24: Media ComparatorByCostTitle Class

## 11. Create a complete console application in the Aims class



```

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

```

Figure 25: Màn hình chính

### 11.1. Người dùng chọn 1: View store



```

Please choose a number: 0-1-2-3
1
|
--- Media in Stock ---
[Inception | Sci-Fi | $19.99] [The Dark Knight | Action | $24.99] [Interstellar | Sci-Fi | $29.99] [The Great Gatsby | Literature | $9.99]
-----
Options:
-----
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
-----
Please choose a number: 0-1-2-3-4

```

Figure 26: Vào trang View store

### 11.1.1. Người dùng tiếp tục chọn 1: See a media's details

```

Please choose a number: 0-1-2-3-4
1
Enter the title of the media: inception
|
--- Search Results for Title: "inception" ---
Title: Inception
Category: Sci-Fi
Cost: $19.99
Length: 148 minutes
Director: Christopher Nolan
-----
Options:
-----
1. Add to cart
2. Play
0. Back
-----
Please choose a number: 0-1-2

```

Figure 27: See a media's details

```

-----
1. Add to cart
2. Play
0. Back
-----
Please choose a number: 0-1-2
1
The media "Inception" has been added to the cart.
Media added to the cart.
Options:
-----
1. Add to cart
2. Play
0. Back
-----
Please choose a number: 0-1-2

```

Figure 28: Thêm vào Cart

## 11.1.2. Người dùng chọn 2: Add a media to the cart

```

-----
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
-----
Please choose a number: 0-1-2-3-4
2
Enter the title of the media to add to the cart: inception

The media "Inception" has been added to the cart.
Media added to the cart.
Options:
-----
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
-----
Please choose a number: 0-1-2-3-4

```

Figure 29: Thêm media vào Cart

## 11.1.3. Người dùng chọn 3: Play a media

```

-----
0. Back
-----
Please choose a number: 0-1-2-3-4
3
Enter the title of the media to play: inception
Playing DVD: Inception
DVD length: 148 minutes
Options:
-----
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
-----
Please choose a number: 0-1-2-3-4

```

Figure 29: Play a media

## 11.1.4. Người dùng chọn 4: See current cart

```
-----
Please choose a number: 0-1-2-3-4
4

--- Cart Contents ---
Title: Inception
Category: Sci-Fi
Cost: $19.99
Length: 148 minutes
Director: Christopher Nolan

-----
Total cost: $19.99
Total items in cart: 1
Options:
-----
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
-----
Please choose a number: 0-1-2-3-4-5
1
Choose filter option: 1. By ID 2. By Title
2
Enter the title to filter by: inception

--- Filter Results by Title: inception ---
Title: Inception
Category: Sci-Fi
Cost: $19.99
Length: 148 minutes
Director: Christopher Nolan
Options:
-----
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
-----
Please choose a number: 0-1-2-3-4-5
```

Figure 30: See current cart after filter

## 11.2. Người dùng chọn 2: Update store

### 11.2.1. Người dùng chọn 1: Add a media to the store

```
AIMS:
-----
1. View store
2. Update store
3. See current cart
0. Exit
-----
Please choose a number: 0-1-2-3
2
Enter the operation: 1 to add, 2 to remove
1
Enter media type (DVD, Book, CD): DVD
Enter title: hellboy

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

Figure 31: Add a media to store

```
[hellboy | Unknown | $19.99]
```

Figure 32: Kết quả sau khi thêm

### 11.2.2. Người dùng chọn 2: Remove a media from the store

```
Media added to the store: Rock Classics
AIMS:
-----
1. View store
2. Update store
3. See current cart
0. Exit
-----
Please choose a number: 0-1-2-3
2
Enter the operation: 1 to add, 2 to remove
2
Enter the title of the media to remove: inception

Media removed from the store: Inception
AIMS:
-----
1. View store
2. Update store
3. See current cart
0. Exit
-----
Please choose a number: 0-1-2-3
1
|
--- Media in Stock ---
[The Dark Knight | Action | $24.99] [Interstellar | Sci-Fi | $29.99]
-----
Options:
-----
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
-----
Please choose a number: 0-1-2-3-4
```

Figure 33: Remove a media from the store and result

### 11.3. Người dùng chọn 3: See current cart

```
'
--- Cart Contents ---
Title: The Dark Knight
Category: Action
Cost: $24.99
Length: 152 minutes
Director: Christopher Nolan
Title: Interstellar
Category: Sci-Fi
Cost: $29.99
Length: 169 minutes
Director: Christopher Nolan
Book ID: 4
Title: The Great Gatsby
Category: Literature
Cost: 9.99
Authors: []
```

Figure 34: Trong cart hiện tại

### 11.3.1. Người dùng chọn 1: Filter medias in cart

```
Total cost: $64.97
Total items in cart: 3
Options:
-----
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
-----
Please choose a number: 0-1-2-3-4-5
1
Choose filter option: 1. By ID 2. By Title
2
Enter the title to filter by: the dark knight
|
--- Filter Results by Title: the dark knight ---
Title: The Dark Knight
Category: Action
Cost: $24.99
Length: 152 minutes
Director: Christopher Nolan
```

Figure 35: Filter cart by Title

### 11.3.2. Người dùng chọn 2: Sort medias in cart

```

-----
Please choose a number: 0-1-2-3-4-5
2
Choose sorting option: 1. By Title 2. By Cost
1

The cart has been sorted by title and then by cost.
Options:
-----
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
-----
Please choose a number: 0-1-2-3-4-5
0
Options:
-----
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
-----
Please choose a number: 0-1-2-3-4
4
|
--- Cart Contents ---
Title: Interstellar
Category: Sci-Fi
Cost: $29.99
Length: 169 minutes
Director: Christopher Nolan
Title: The Dark Knight
Category: Action
Cost: $24.99
Length: 152 minutes
Director: Christopher Nolan
Book ID: 4
Title: The Great Gatsby
Category: Literature
Cost: 9.99
Authors: []

```

Figure 36: Sort cart by Title

### 11.3.3. Người dùng chọn 3: Remove media from cart

```
-----
Please choose a number: 0-1-2-3-4-5
3
Enter the title of the media to remove: The dark knight

The media "The Dark Knight" has been removed from the cart.
Options:
-----
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
-----
Please choose a number: 0-1-2-3-4-5
0
Options:
-----
1. See a media's details
2. Add a media to cart
3. Play a media
4. See current cart
0. Back
-----
Please choose a number: 0-1-2-3-4
4
|
--- Cart Contents ---
Title: Interstellar
Category: Sci-Fi
Cost: $29.99
Length: 169 minutes
Director: Christopher Nolan
Book ID: 4
Title: The Great Gatsby
Category: Literature
Cost: 9.99
Authors: []
```

Figure 37: Remove media from cart

#### 11.3.4. Người dùng chọn 4: Play a media



```
Total cost: $59.90
Total items in cart: 2
Options:
-----
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
-----
Please choose a number: 0-1-2-3-4-5
4
Enter the title of the media to play: Interstellar
Playing DVD: Interstellar
DVD length: 169 minutes
```

Figure 38: Play a media

### 11.3.5. Người dùng chọn 5: Place order

```
0. Back
-----
Please choose a number: 0-1-2-3-4-5
5

Order placed successfully with destination
Order placed and cart is now empty.
```

Figure 39: Place order

## 12. Class diagram

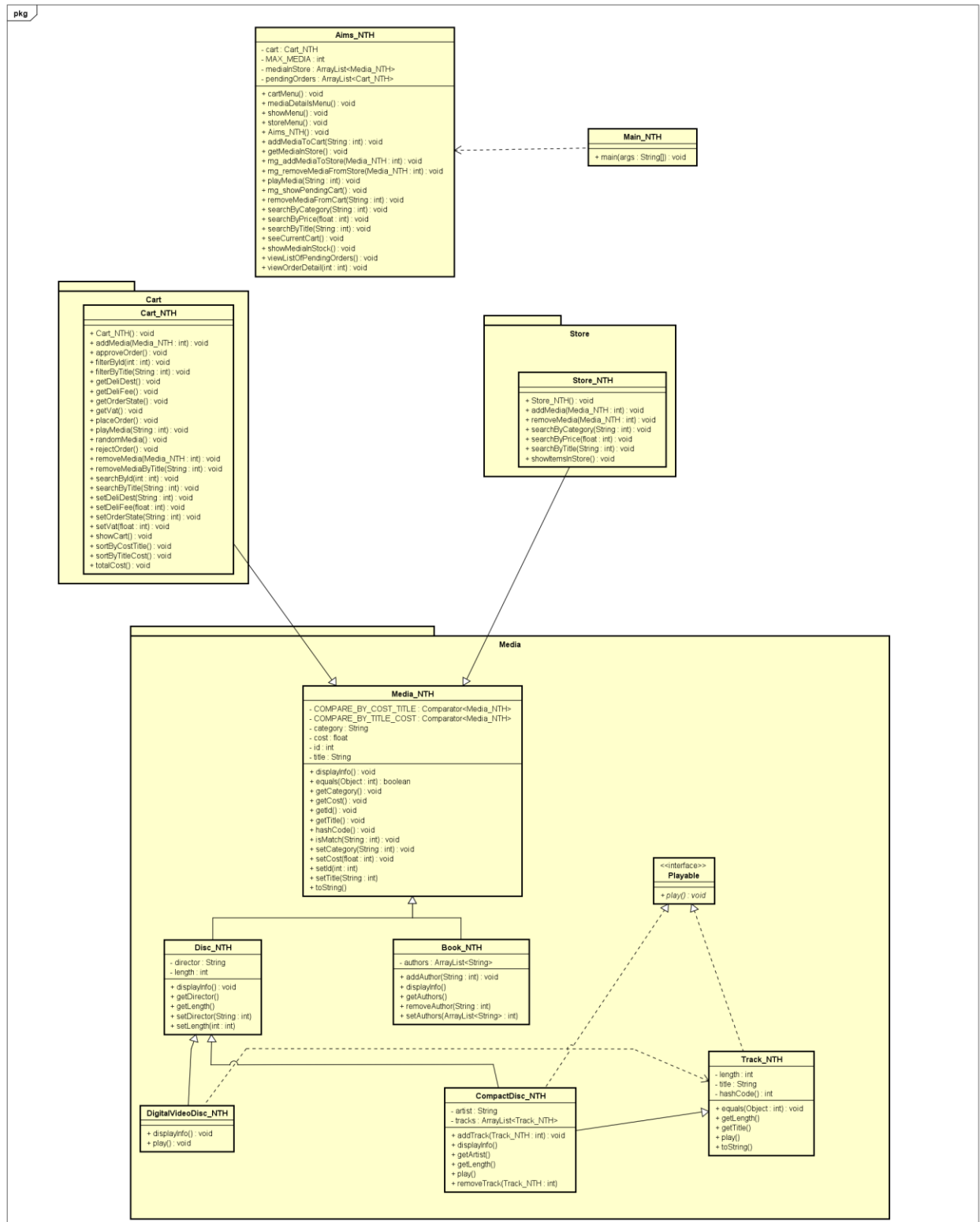


Figure 40: Class diagram

## 13. UseCase diagram

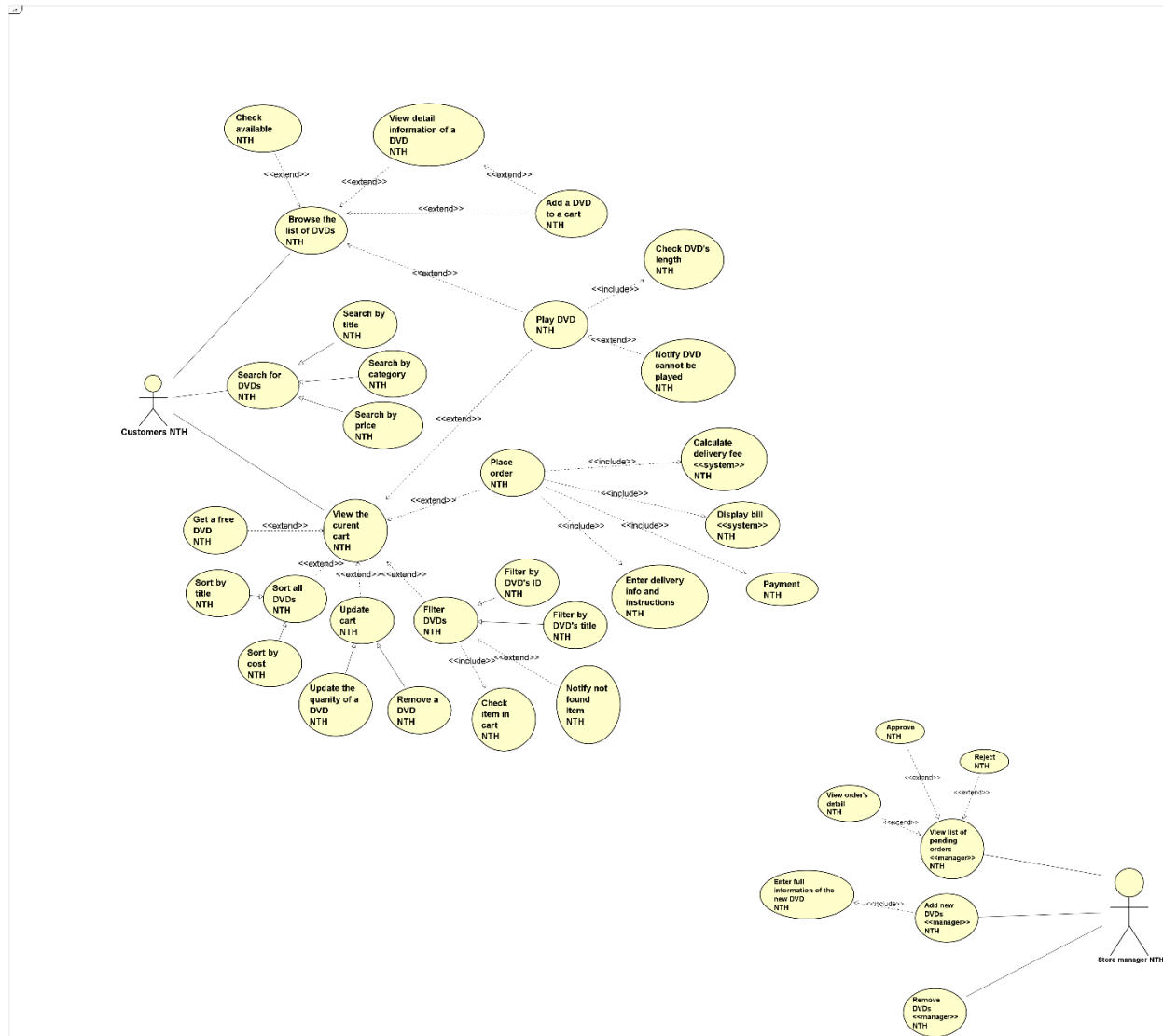


Figure 41: UseCase diagram

## 14. Answer Question

### - Lớp nào nên implement interface Comparable?

Lớp Media\_NTH hoặc các lớp con của nó (ví dụ: DVD, Book, ...) nên implement interface Comparable. Lý do là interface Comparable cung cấp một phương thức compareTo() dùng để so sánh các đối tượng của lớp. Lớp Media\_NTH là lớp cha của tất cả các loại media, vì vậy ta có thể implement Comparable trong lớp này để đảm bảo rằng mọi đối tượng của các lớp con như

DVD, Book sẽ có thể so sánh được. Nếu cần quy tắc so sánh riêng biệt cho mỗi loại media, ta có thể override phương thức `compareTo()` trong từng lớp con.

- **Cách cài đặt phương thức `compareTo()` để phản ánh thứ tự mà ta muốn?**

Phương thức `compareTo()` cần xác định rõ quy tắc so sánh giữa các đối tượng. Khi cài đặt `compareTo()`, ta phải so sánh các thuộc tính của đối tượng để quyết định thứ tự của chúng. Quy tắc so sánh này có thể là so sánh theo một hoặc nhiều thuộc tính, tùy thuộc vào yêu cầu. Phương thức `compareTo()` sẽ trả về một giá trị âm, dương hoặc bằng 0 tùy theo việc đối tượng này có nhỏ hơn, lớn hơn hay bằng đối tượng còn lại.

Nếu ta muốn sắp xếp theo một quy tắc cụ thể (ví dụ, theo tên, sau đó là giá), ta sẽ triển khai phương thức này để đầu tiên so sánh theo tên, và nếu tên giống nhau thì tiếp tục so sánh theo giá.

- **Liệu có thể có hai quy tắc sắp xếp (theo tiêu đề rồi đến giá, và theo giá rồi đến tiêu đề) khi dùng interface `Comparable`?**

Không, mỗi lớp chỉ có thể có một quy tắc sắp xếp duy nhất khi implement interface `Comparable`. Điều này có nghĩa là chỉ có một cách so sánh mặc định cho các đối tượng của lớp đó. Nếu cần hai quy tắc sắp xếp khác nhau, ví dụ như một quy tắc sắp xếp theo tiêu đề rồi đến giá và một quy tắc sắp xếp theo giá rồi đến tiêu đề, ta phải sử dụng `Comparator` thay vì `Comparable`. `Comparator` cho phép định nghĩa nhiều quy tắc sắp xếp khác nhau mà không ảnh hưởng đến quy tắc sắp xếp mặc định của lớp.

- **Nếu các DVD có quy tắc sắp xếp khác biệt (theo tiêu đề, sau đó là chiều dài giảm dần, và cuối cùng là giá), ta sẽ làm như thế nào?**

Để quy tắc sắp xếp riêng biệt cho DVD, ta cần override phương thức `compareTo()` trong lớp DVD. Điều này có nghĩa là lớp DVD sẽ có phương thức `compareTo()` riêng để thực hiện các phép so sánh theo tiêu đề, chiều dài giảm dần và giá. Quy tắc sắp xếp của lớp DVD sẽ không bị ảnh hưởng bởi quy tắc sắp xếp mặc định trong lớp `Media_NTH`.

Lớp `Media_NTH` có thể vẫn giữ phương thức `compareTo()` mặc định cho các loại media khác, trong khi lớp `DVD` sẽ cài đặt lại phương thức `compareTo()` để xử lý việc so sánh theo quy tắc riêng của `DVD`.