

Object-Oriented Programming

Lab 04: Inheritance and Polymorphism

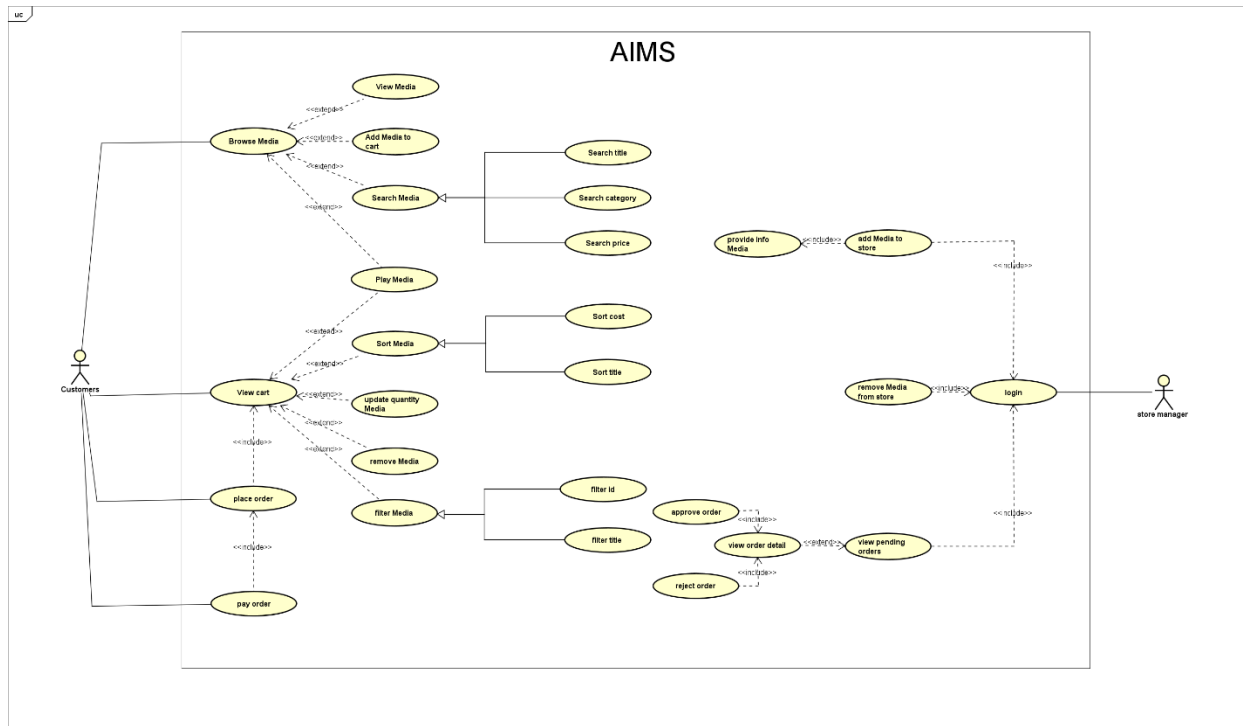
Họ và Tên: Nguyễn Bá Hoàng

MSSV: 20225844

Mã Lớp: 744520

1.Import the existing project into the workspace of Eclipse

2.Additional requirements of AIMS



3.Creating the Book class

```
1 package hust.soict.hedspi.aims.media;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class Book extends Media { 11 usages
7     private List<String> authors = new ArrayList<String>(); 4 usages
8
9     public Book(int id, String title, String category, float cost) { 5 usages
10         super(id, title, category, cost);
11     }
12
13     public void addAuthor(String authorName){ 4 usages
14         if(authors.contains(authorName) == false){
15             authors.add(authorName);
16         }else{
17             System.out.println("da ton tai author");
18         }
19     }
20
21     public void removeAuthor(String authorName){ no usages
22         if(authors.contains(authorName) == true){
23             authors.remove(authorName);
24         }else{
25             System.out.println("khong ton tai author de xoa");
26         }
27     }
28
29     public String toString() {
30         String Book_print = this.getTitle();
31
32         if(this.getCategory() != null){
```

```
29     public String toString() {
30         String Book_print = this.getTitle();
31
32         if(this.getCategory() != null){
33             Book_print += " - ";
34             Book_print += this.getCategory();
35         }
36
37         return "Book - " + Book_print + ": " + this.getCost() + "$";
38     }
39
40 }
41 }
```

4. Creating the abstract Media class

```
Book.java  Media.java x
1 package hust.soiect.hedspi.aims.media;
2
3
4 import java.util.Comparator;
5
6 public abstract class Media implements Comparable<Media> { 59 usages 4 inheritors
7     private int id; 2 usages
8     private String title; 6 usages
9     private String category; 2 usages
10    private float cost; 4 usages
11
12    public static final Comparator<Media> COMPARE_BY_TITLE_COST = new MediaComparatorByTitleCost(); 2 usages
13    public static final Comparator<Media> COMPARE_BY_COST_TITLE = new MediaComparatorByCostTitle(); 2 usages
14
15    public Media(int id, String title, String category, float cost) { 2 usages
16        this.id = id;
17        this.title = title;
18        this.category = category;
19        this.cost = cost;
20    }
21
22    public int getId() { 1 usage
23        return id;
24    }
25
26    public String getTitle() {
27        return title;
28    }
29
30    public String getCategory() { 6 usages
31        return category;
32    }
33
34    public float getCost() { 10 usages
35        return cost;
36    }
37
38
39    boolean equals(Media mediaCheck){ no usages
40        if(mediaCheck == null){
41            return false;
42        }
43
44        if(this.title == mediaCheck.title){
45            return true;
46        }else{
47            return false;
48        }
49    }
50
51    //compareTo theo title
52    @Override 1 override
53    public int compareTo(Media o){
54        int compareTitle = this.title.compareTo(o.title);
55        if(compareTitle == 0){
56            int compareCost = Float.compare(o.cost, this.cost);
57            return compareCost;
58        }
59        return compareTitle;
60    }
61
```

```

62
63     //compareTo theo cost
64     /*
65     @Override
66     public int compareTo(Media o){
67         int compareCost = Float.compare(o.cost,this.cost);
68
69         if(compareCost == 0){
70             int compareTitle = this.title.compareTo(o.title);
71             return compareTitle;
72         }
73         return compareCost;
74     }
75     */
76
77
78     public String toString() { 3 overrides
79         String Media_print = this.getTitle();
80
81         if(this.getCategory() != null){
82             Media_print += " - ";
83             Media_print += this.getCategory();
84         }
85
86         return "Media - " + Media_print + ": " + this.getCost() + "$";
87     }
88
89 }

```

5. Creating the CompactDisc class

5.1. Create the Disc class extending the Media class

```

1 package hust.soict.hedspi.aims.media;
2
3 @ public class Disc extends Media { 2 usages 2 inheritors
4     private String director; 2 usages
5     private int length = 0; 2 usages
6
7     public Disc(int id, String title, String category, float cost, String director, int length) { 2 usages
8         super(id, title, category, cost);
9         this.director = director;
10        this.length = length;
11    }
12
13    public String getDirector() { 4 usages
14        return director;
15    }
16
17    public int getLength() { 5 usages
18        return length;
19    }
20 }
21

```

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

```
Disc.java  Track.java  CompactDisc.java
1 package hust.soict.hedspi.aims.media;
2
3 public class Track implements Playable { 26 usages
4     private String title; 4 usages
5     private int length; 4 usages
6
7     public Track(String title,int length) { 9 usages
8         this.title = title;
9         this.length = length;
10    }
11
12    public String getTitle() {
13        return title;
14    }
15
16    public int getLength() { 4 usages
17        return length;
18    }
19
20    public void play() { 7 usages
21        System.out.println("Playing Track: " + this.getTitle());
22        System.out.println("Track length: " + this.getLength());
23    }
24
25
26    boolean equals(Track trackCheck){ no usages
27        if(trackCheck == null){
28            return false;
29        }
30
31        if(this.title == trackCheck.title){
32            if(this.length == trackCheck.length){
```

```
32            if(this.length == trackCheck.length){
33                return true;
34            }else {
35                return false;
36            }
37        }else {
38            return false;
39        }
40    }
41
42 }
43
```

5.3. Open the CompactDisc class

```
1 package hust.soict.hedspi.aims.media;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class CompactDisc extends Disc implements Playable { 20 usages
7     private String artist; 5 usages
8     private List<Track> tracks = new ArrayList<Track>(); 7 usages
9
10    public CompactDisc(int id, String title, String category, float cost, String director, String artist) { 5 usages
11        super(id, title, category, cost, director, length: 0);
12        this.artist = artist;
13    }
14
15    public String getArtist() { no usages
16        return artist;
17    }
18
19    public void addTrack(Track tracksInput){ 9 usages
20        for(Track tmp :tracks){
21            if(tmp != null) {
22                if (tmp.getTitle() == tracksInput.getTitle()) {
23                    if(tmp.getLength() == tracksInput.getLength()) {
24                        System.out.println("track da ton tai");
25                        return;
26                    }
27                }
28            }
29        }
30        tracks.add(tracksInput);
31    }
32}
```



```

32
33     public void removeTrack(Track tracksInput){ no usages
34         for(Track tmp :tracks){
35             if(tmp != null) {
36                 if (tmp.getTitle() == tracksInput.getTitle()) {
37                     tracks.remove(tmp);
38                     return;
39                 }
40             }
41         }
42         System.out.println("track khong ton tai");
43     }
44
45
46     public int getLength_CD(){ 3 usages
47         int count_length =0;
48         for(Track tmp :tracks) {
49             if (tmp != null) {
50                 count_length += tmp.getLength();
51             }
52         }
53         return count_length;
54     }
55

```

```

56 ⑥↑     public void play() { 7 usages
57         System.out.println("Playing CD: " + this.getTitle());
58         System.out.println("artist: " + artist);
59         System.out.println("Number track in CD: " + tracks.size());
60
61         System.out.println("CD length: " + this.getLength_CD());
62
63         //play each tracks
64         for(Track tmp :tracks){
65             if(tmp != null){
66                 tmp.play();
67             }
68         }
69
70     }
71

```

```

73  public String toString() {
74      String CD_print = this.getTitle();
75
76      if(this.getCategory() != null){
77          CD_print += " - ";
78          CD_print += this.getCategory();
79      }
80
81      if(this.getDirector() != null){
82          CD_print += " - ";
83          CD_print += this.getDirector();
84      }
85
86      if(this.getLength_CD() != 0){
87          CD_print += " - ";
88          CD_print += this.getLength_CD();
89      }
90
91      if(this.artist != null){
92          CD_print += " - ";
93          CD_print += this.artist;
94      }
95
96      return "CD - |" + CD_print + ": " + this.getCost() + "$";
97  }
98
99  }

```

6. Create the Playable interface

-Playable interface:

```
1      package hust.soict.hedspi.aims.media;
2
3      public interface Playable { 3 usages 3 implementations
4          public void play(); 7 usages 3 implementations
5      }
6      |
```

- DigitalVideoDisc class:

```
42      public void play() { 7 usages new *
43          System.out.println("Playing DVD: " + this.getTitle());
44          System.out.println("DVD length: " + this.getLength());
45      }
46
```

- CompactDisc class:

```

56 ①↑ public void play() { 7 usages
57      System.out.println("Playing CD: " + this.getTitle());
58      System.out.println("artist: " + artist);
59      System.out.println("Number track in CD: " + tracks.size());
60
61      System.out.println("CD length: " + this.getLength_CD());
62
63      //play each tracks
64      for(Track tmp :tracks){
65          if(tmp != null){
66              tmp.play();
67          }
68      }
69
70  }

```

- Track class:

```

20 ①↑ public void play() { 7 usages
21      System.out.println("Playing Track: " + this.getTitle());
22      System.out.println("Track length: " + this.getLength());
23  }
24

```

7. Update the Cart class to work with Media

```
1 package hust.soict.hedspi.aims.Cart;
2
3 import hust.soict.hedspi.aims.media.Media;
4
5 import java.util.ArrayList;
6 import java.util.Collections;
7
8 public class Cart { 6 usages  🡕 hello24680 *
9
10     private ArrayList<Media> itemsOrdered = new ArrayList<Media>(); 13 usages
11
12     public void addMedia(Media mediaNew){ 2 usages  🡕 hello24680 *
13         for(Media tmp :itemsOrdered){
14             if(tmp != null) {
15                 if (tmp.getTitle() == mediaNew.getTitle()) {
16                     System.out.println("media da ton tai");
17                     return;
18                 }
19             }
20         }
21         itemsOrdered.add(mediaNew);
22     }
23
24     public void removeMedia(Media mediaRemove){ 1 usage  new *
25         for(Media tmp :itemsOrdered){
26             if(tmp != null) {
27                 if (tmp.getTitle() == mediaRemove.getTitle()) {
28                     itemsOrdered.remove(tmp);
29                     return;
30                 }
31             }
32         }
```

```
33     System.out.println("Media khong ton tai");
34 }
35
36 public float totalCost(){ 1 usage  🡕 hello24680 *
37     float count_cost = 0;
38     for(Media tmp :itemsOrdered){
39         if(tmp != null) {
40             count_cost += tmp.getCost();
41         }
42     }
43     return count_cost;
44 }
45
46 public void sortByTitle(){ 1 usage  new *
47     Collections.sort(itemsOrdered, Media.COMPARE_BY_TITLE_COST);
48     for(Media tmp: itemsOrdered){
49         if(tmp != null){
50             System.out.println(tmp.toString());
51         }
52     }
53 }
54
55 public void sortByCost() { 1 usage  🡕 hello24680 *
56     Collections.sort(itemsOrdered, Media.COMPARE_BY_COST_TITLE);
57     for(Media tmp: itemsOrdered){
58         if(tmp != null){
59             System.out.println(tmp.toString());
60         }
61     }
62 }
```

```

62     }
63
64     public void printCart() { 3 usages  🡕 hello24680 *
65         System.out.println("*****CART*****");
66         System.out.println("Ordered Items:");
67
68         for (Media tmp : itemsOrdered) {
69             System.out.println(tmp.toString());
70         }
71
72         System.out.println("Total cost: " + totalCost() + "$");
73         System.out.println("*****");
74     }
75
76     public Media searchByTitle(String title) { 3 usages  🡕 hello24680 *
77         for (Media tmp : itemsOrdered) {
78             if (tmp.getTitle().equals(title))
79                 return tmp;
80         }
81
82         return null;
83     }
84
85     public Media searchById(int id) { 1 usage  🡕 hello24680 *
86         for (Media tmp : itemsOrdered) {
87             if (tmp.getId() == id)
88                 return tmp;
89         }
90
91         return null;
92     }

```

```

92     }
93
94     public void empty() { 1 usage  new *
95         itemsOrdered.clear();
96     }
97 }
98

```

8. Update the Store class to work with Media

```
public class Store { 6 usages  🡕 hello24680 *
    private ArrayList<Media> itemsInStore = new ArrayList<Media>(); 6 usages

    public void addMedia(Media mediaNew){ 12 usages  🡕 hello24680 *
        for(Media tmp :itemsInStore){
            if(tmp != null) {
                if (tmp.getTitle() == mediaNew.getTitle()) {
                    System.out.println("media da ton tai");
                    return;
                }
            }
        }
        itemsInStore.add(mediaNew);
    }

    public void removeMedia(Media mediaRemove){ 1 usage  🡕 hello24680 *
        for(Media tmp :itemsInStore){
            if(tmp != null) {
                if (tmp.getTitle() == mediaRemove.getTitle()) {
                    itemsInStore.remove(tmp);
                    return;
                }
            }
        }
        System.out.println("Media khong ton tai");
    }
}
```



```














public void printStore() { 6 usages  👤 hello24680 *
    for (Media tmp : itemsInStore) {
        System.out.println(tmp.toString());
    }
}

public Media searchByTitle(String title) { 4 usages  👤 hello24680 *
    for (Media tmp : itemsInStore) {
        if (tmp.getTitle().equals(title)) {
            return tmp;
        }
    }
    return null;
}
}

```

9. Constructors of whole classes and parent classes

- Cấu trúc project

- ▼  src
 - ▼  hust.soict.hedspi
 - ▼  aims
 - ▼  Cart
 - © Cart
 - ▼  media
 - © Book
 - © CompactDisc
 - © DigitalVideoDisc
 - © Disc
 - © Media
 - © MediaComparatorByCostTitle
 - © MediaComparatorByTitleCost
 - ① Playable
 - © Track
 - ▼  store
 - © Store
 - 🔄 Aims
 - ▼  test
 - >  cart
 - ▼  media
 - © BookTest
 - 🔄 MediaTest
 - 🔄 TestPassingParameter
 - >  store
 -  AimsProject.iml
 -  Report.pdf
 - >  External Libraries

```

classDiagram
    class Aims {
        +maintain: String[] void
        +deleteShow() void
        +showMenu() void
        +storeMenu() void
        +mediaDetailsView(media: Media) void
        +confirm() void
        +updateStore() void
    }
    class Cart {
        +addMedia(media: Media) void
        +removeMedia(media: Media) void
        +totalCost() float
        +sortByTitle() void
        +sortByCost() void
        +printCart() void
        +searchByTitle(title: String) Media
        +searchById(id: Media) Media
        +empty() void
    }
    class Scanner {
    }
    class Store {
        +addMedia(media: Media) void
        +removeMedia(media: Media) void
        +printStore() void
        +searchByTitle(title: String) Media
    }
    class Media {
        +id: int
        +title: String
        +category: String
        +cost: float
        +Media(id: int, title: String, category: String, cost: float)
        +getTitle() String
        +getCategory() String
        +getCost() float
        +equals(media: Media) boolean
        +compareTo(media: Media) int
    }
    class Disc {
        +director: String
        +length: int
        +Disc(id: int, title: String, category: String, cost: float, director: String, length: int)
        +getDirector() String
        +getLength() int
    }
    class Book {
        +authors: List<String> = new ArrayList<>()
        +Book(id: int, title: String, category: String, cost: float)
        +addAuthor(authorName: String) void
        +removeAuthor(authorName: String) void
        +toString() String
    }
    class DigitalVideoDisc {
        +DigitalVideoDisc(id: int, title: String, category: String, cost: float, director: String, length: int)
        +getTitle() String
        +isAvailable() boolean
        +play() void
        +compareTo(media: Media) int
    }
    class CompactDisc {
        +artist: String
        +CompactDisc(id: int, title: String, category: String, cost: float, director: String, artist: String)
        +getArtist() String
        +addTrack(track: Track) void
        +removeTrack(track: Track) void
        +getLength() int
        +play() void
        +toString() String
    }
    class Track {
        +title: String
        +length: int
        +Track(title: String, length: int)
        +getTitle() String
        +getLength() int
        +play() void
        +equals(track: Track) boolean
    }
    class Comparable {
        <<interface>>
    }
    class List {
        <<interface>>
    }
    class ComparableMedia {
        <<interface>>
    }
    class Comparator {
        <<interface>>
    }
    class Playable {
        <<interface>>
    }
    class ComparatorMedia {
        <<interface>>
    }
    class MediaComparatorByCost {
        +compare(o1: Media, o2: Media) int
    }
    class MediaComparatorByTitle {
        +compare(o1: Media, o2: Media) int
    }

    Aims --> Cart : cart
    Aims --> Scanner : scanner
    Aims --> Store : store
    Cart --> Media : addMedia
    Cart --> Media : removeMedia
    Scanner --> Media : searchByTitle
    Scanner --> Media : searchById
    Store --> Media : addMedia
    Store --> Media : removeMedia
    Store --> Media : searchByTitle
    Media --> Disc : isOrdered
    Media --> Book : isOrdered
    Media --> DigitalVideoDisc : isOrdered
    Media --> CompactDisc : isOrdered
    Media --> Track : isOrdered
    Disc --|> DigitalVideoDisc
    Disc --|> CompactDisc
    Disc --|> Track
    Book --|> DigitalVideoDisc
    Book --|> CompactDisc
    Book --|> Track
    DigitalVideoDisc --|> Track
    CompactDisc --|> Track
    ComparableMedia <|-- Media
    ComparatorMedia <|-- MediaComparatorByCost
    ComparatorMedia <|-- MediaComparatorByTitle
    Playable <|-- Media
    
```

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

Trả lời: + media là cha của disc và book; disc là cha của DVDs và CD.

+ các constructor đã được khởi tạo như dưới đây:

- Constructors của các lớp cha và con:

Media:

```
public Media(int id, String title, String category, float cost) { 2 usages
    this.id = id;
    this.title = title;
    this.category = category;
    this.cost = cost;
}
```

Disc:

```
public Disc(int id, String title, String category, float cost, String director, int length) { 2
    super(id, title, category, cost);
    this.director = director;
    this.length = length;
}
```

Book:

```
public Book(int id, String title, String category, float cost) { 50
    super(id, title, category, cost);
}
```

DVDs:

```
public DigitalVideoDisc(int id, String title, String category, float cost, String director, int length) {
    super(id, title, category, cost, director, length);
}
```

CD:

```
public CompactDisc(int id, String title, String category, float cost, String director, String artist) {
    super(id, title, category, cost, director, length: 0);
    this.artist = artist;
}
```

10. Unique item in a list

- Media class:

```
boolean equals(Media mediaCheck){ no usage
    if(mediaCheck == null){
        return false;
    }

    if(this.title == mediaCheck.title){
        return true;
    }else{
        return false;
    }
}
```

- Track class:

```

boolean equals(Track trackCheck){  no usages
    if(trackCheck == null){
        return false;
    }

    if(this.title == trackCheck.title){
        if(this.length == trackCheck.length){
            return true;
        }else {
            return false;
        }
    }else {
        return false;
    }
}

```

Câu hỏi: If the passing object is not an instance of Media, what happens?

Trả lời: **Nếu đối tượng truyền vào không phải là một instance của Media hoặc không phải con của Media thì sẽ không thực hiện được phép so sánh và java sẽ báo lỗi tại nơi truyền parameter vào phương thức.**

11. Polymorphism with toString() method

-test:

```

public static void main(String[] args){
    List<Media> mediae = new ArrayList<Media>();

    //create media
    //CD
    CompactDisc CD1 = new CompactDisc( id: 1, title: "MU", category: "1 mon qua", cost: 1.5f, director: "siu", artist: "hoang1");
    //dvd
    DigitalVideoDisc DVD1 = new DigitalVideoDisc( id: 2, title: "Barsa", category: "null", cost: 0.5f, director: "siuu", length: 1);
    //book
    Book Book3 = new Book( id: 3, title: "Che", category: "null", cost: 3f);

    //add vao mediae
    mediae.add(CD1);
    mediae.add(DVD1);
    mediae.add(Book3);

    //test in ra man hinh
    System.out.println("\n");
    System.out.println("test add");
    for(Media m: mediae){
        System.out.println(m.toString());
    }
}

```

-ket qua:

```

test add
CD - MU - 1 mon qua - siu - hoang1: 1.5$
DVD - Barsa - null - siuu - 1: 0.5$
Book - Che - null: 3.0$

Process finished with exit code 0

```

-Nx:toString() được override để hiển thị chi tiết các thuộc tính trong các lớp CD, DVD và Book của media cha. Do đó java sẽ sử dụng toString() trong các lớp cd,dvd,book thay vì của media.

-media:

```
public String toString() { 3 overrides
    String Media_print = this.getTitle();

    if(this.getCategory() != null){
        Media_print += " - ";
        Media_print += this.getCategory();
    }

    return "Media - " + Media_print + ": " + this.getCost() + "$";
}
```

-cd:


```

public String toString() {
    String CD_print = this.getTitle();

    if(this.getCategory() != null){
        CD_print += " - ";
        CD_print += this.getCategory();
    }

    if(this.getDirector() != null){
        CD_print += " - ";
        CD_print += this.getDirector();
    }

    if(this.getLength_CD() != 0){
        CD_print += " - ";
        CD_print += this.getLength_CD();
    }

    if(this.artist != null){
        CD_print += " - ";
        CD_print += this.artist;
    }

    return "CD - " + CD_print + ": " + this.getCost() + "$";
}

```

-dvd:

```

public String toString() {
    String dvd_print = this.getTitle();

    if(this.getCategory() != null){
        dvd_print += " - ";
        dvd_print += this.getCategory();
    }

    if(this.getDirector() != null){
        dvd_print += " - ";
        dvd_print += this.getDirector();
    }

    if(this.getLength() != 0){
        dvd_print += " - ";
        dvd_print += this.getLength();
    }
    return "DVD - " + dvd_print + ": " + this.getCost() + "$";
}

```

-book:

```

public String toString() {
    String Book_print = this.getTitle();

    if(this.getCategory() != null){
        Book_print += " - ";
        Book_print += this.getCategory();
    }

    return "Book - " + Book_print + ": " + this.getCost() + "$";
}

```

12. Sort media in the cart

MediaComparatorByCostTitle:

```
1 package hust.soict.hedspi.aims.media;
2
3 import java.util.Comparator;
4
5 public class MediaComparatorByCostTitle implements Comparator<Media> { 1 usage
6     @Override
7     public int compare(Media o1, Media o2) {
8         int compareCost = Float.compare(o2.getCost(), o1.getCost());
9
10        if(compareCost == 0){
11            int compareTitle = o1.getTitle().compareTo(o2.getTitle());
12            return compareTitle;
13        }
14        return compareCost;
15    }
16 }
17
```

MediaComparatorByTitleCost:

```
1 package hust.soict.hedspi.aims.media;
2
3 import java.util.Comparator;
4
5 public class MediaComparatorByTitleCost implements Comparator<Media> { 1 usage
6     public int compare(Media o1, Media o2) {
7         int compareTitle = o1.getTitle().compareTo(o2.getTitle());
8
9         if(compareTitle == 0){
10            int compareCost = Float.compare(o2.getCost(), o1.getCost());
11            return compareCost;
12        }
13        return compareTitle;
14    }
15 }
16
```

Kết quả:

```
compare by title
DVD - Barsa - null - siuu - 1: 0.5$
Book - Che - null: 3.0$
CD - MU - 1 mon qua - siu - hoang1: 1.5$

compare by cost
Book - Che - null: 3.0$
CD - MU - 1 mon qua - siu - hoang1: 1.5$
DVD - Barsa - null - siuu - 1: 0.5$

compareTo by title
DVD - Barsa - null - siuu - 1: 0.5$
Book - Che - null: 3.0$
CD - MU - 1 mon qua - siu - hoang1: 1.5$

compareTo DVD theo title->length->cost
DVD - Barsa - null - siuu - 2: 2.0$
DVD - Barsa - null - siuu - 2: 0.5$
DVD - Barsa - null - siuu - 1: 0.5$
Book - Che - null: 3.0$
CD - MU - 1 mon qua - siu - hoang1: 1.5$
|
Process finished with exit code 0
```

Câu hỏi:

a. What class should implement the Comparable interface?

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

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

-triển khai như trên đây:

```
//compareTo theo title
@Override
public int compareTo(Media o){
    int compareTitle = this.title.compareTo(o.title);
    if(compareTitle == 0){
        int compareCost = Float.compare(o.cost, this.cost);
        return compareCost;
    }
    return compareTitle;
}

//compareTo theo cost
/*
@Override
public int compareTo(Media o){
    int compareCost = Float.compare(o.cost, this.cost);

    if(compareCost == 0){
        int compareTitle = this.title.compareTo(o.title);
        return compareTitle;
    }
    return compareCost;
}
*/
```

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

- Không ta không thể, Comparable chỉ cho phép định nghĩa một quy tắc sắp xếp duy nhất thông qua phương thức compareTo().

Nếu cần nhiều quy tắc thì khi đó ta phải sử dụng Comparator như đã triển khai trong MediaComparatorByCostTitle và MediaComparatorByTitleCost.

d. 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?

-em đã làm như trên đây:

```
//compareTo DVD theo title->length->cost
@Override new *
public int compareTo(Media other) {
    if (other instanceof DigitalVideoDisc) {
        DigitalVideoDisc otherDVD = (DigitalVideoDisc) other;

        // So sánh title
        int titleComparison = this.getTitle().compareTo(otherDVD.getTitle());
        if (titleComparison != 0) {
            return titleComparison;
        }

        // So sánh length
        int lengthComparison = Integer.compare(otherDVD.getLength(), this.getLength());
        if (lengthComparison != 0) {
            return lengthComparison;
        }

        // So sánh cost
        return Float.compare(otherDVD.getCost(), this.getCost());
    }

    // các loại khác sử dụng quy tắc của lớp cha media
    return super.compareTo(other);
}
```

13. Create a complete console application in the Aims class

(do aims class dài hơn 500 dòng và nhiều tổ hợp kiểm tra nên sẽ không viết ở đây mà để code trên git để mọi người check ạ)