Report Lab04
Student's name: Le Viet Cuong
Student's id: 20235586

9. Constructors of whole classes and parent classes
Question: Which classes are aggregates of other classes? Checking all constructors of whole classes if they initialize for their parts?
Answer:
- **Store** aggregates **Media**.
- **Cart** aggregates **Media**.
- **CompactDisc** aggregates **Track**.

10. Unique item in the list
Question: If the passing object is not an instance of Media, what happens?
Answer: If the passing object is not an instance of Media, the method **Equals** will return **false** .

```
if(o== null || !(o instanceof Track)) {
    return false;
}
```

11. Polymorphism with toString() method
Question: Create an ArrayList of Media, then add some media (CD, DVD or Book) into the list. Iterate through the list and print out the information of the media by using toString() method. Observe what happens and explain in detail.
Answer:

```
public class TestPassingParameter {

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

        ArrayList<String> authors= new ArrayList<String>();
        authors.add("Volvo");
        authors.add("BMW");
        authors.add("Ford");

        ArrayList<Track> CDs= new ArrayList<Track>();
        Track t1= new Track("Episode 1", 11);
        Track t2= new Track("Episode 2", 2);
        CDs.add(t1);
        CDs.add(t2);

        CompactDisc cd= new CompactDisc(1, "Gaoranger", "Super Sentai", 30.15f, "Kurosaki", "Ichigo", CDs);
        DigitalVideoDisc dvd= new DigitalVideoDisc(2, "Star wars", "Science fiction", 24.95f, 87, "Geogre Lucas");
        Book book= new Book(3, "Murphy", "Psychology", 17.65f, authors);

        mediae.add(cd);
        mediae.add(dvd);
        mediae.add(book);

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

- **Disc** and **Book** inherit **Media**.
- **CompactDisc** and **DigitalVideo Disc** inherit **Disc**.

Media:

```
public abstract class Media {
    public int id;
    public String title;
    public String category;
    public float cost;
```

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

Disc:

```
public String toString() {
    return "Disc - [" + this.getTitle() + "] - [" + this.getCategory()
            + "] - [" + this.getDirector() + "] - [" + this.getLength() + "] : ["
            + this.getCost() + "]$";
}
```

Book:

```
public String toString() {
    return "Book - [" + this.getTitle() + "] - [" + this.getCategory() + "] - ["+ this.authors
            + "] : [" + this.getCost() + "]$";
}
```

CD:

```
public String toString() {
    String tr= "";
    for(Track track: tracks) {
        tr+= "\n+[" + track.getTitle() + "] - [" + track.getLength() + " minutes]";
    }

    return "CD - [" + this.getTitle() + "] - [" + this.getCategory() + "] - [" + this.getDirector()
            + "] - [" + this.getArtist() + "] - [" + this.getLength() + "] : ["
            + this.getCost() + "]$" + tr;
}
```

DVD:

```
public String toString() {
    return "DVD - [" + this.getTitle() + "] - [" + this.getCategory()
            + "] - [" + this.getDirector() + "] - [" + this.getLength() + "] : ["
            + this.getCost() + "]$";
}
```

Result:

```
CD - [Gaoranger] - [Super Sentai] - [Kurosaki] - [Ichigo] - [13] : [30.15]$
+[Episode 1] - [11 minutes]
+[Episode 2] - [2 minutes]
DVD - [Star wars] - [Science fiction] - [Geogre Lucas] - [87] : [24.95]$
Book - [Murphy] - [Psychology] - [[Volvo, BMW, Ford]] : [17.65]$
```

- As we can see:

+ Since **Disc** and **Book** inherit **Media**, those class will also inherit all attributes of Media, including id, tittle, category, and cost. So we don't need to delare them anymore, instead, we can use **getter** to access those attributes.

+ Since **CompactDisc, DigitalVideoDisc** inherit **Disc**, all attributes of **Disc** (including **Media**) will be inherited. Moreover, we **override** the method **toString** in those class, so the output will display the method **toString** of **CompactDisc**, **DigitalVideoDisc**, instead of **Disc**. Similary to **Book**, which inherits **Media**.

12. Sort media in the cart

Question: Alternatively, to compare items in the cart, instead of using Comparator, we can use the Comparable interface and override the compareTo()method. You can refer to the Java docs to see the information of this interface.

Suppose we are taking this Comparable interface approach.

- What class should implement the Comparable interface?
- In those classes, how should you implement the compareTo()method be to reflect the ordering that we want?
- 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?
- 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?

Answer:

- Media should implement the Comparable interface, so we can define our own sorting rule.
- Implement the **compareTo()** method:

```
@Override
public int compareTo(Media o) {
    int comp= this.getTitle().compareTo(o.getTitle());
    if(comp==1) {
        return 1;
    }
    if(this.getCost()< o.getCost()) {
        comp= 1;
    }
    return comp;
}
```

- No, we can't have two ordering rule of the item, since Comparable only define an only default ordering rule. Instead, we can use Comparator.
- We will Override compareTo in DigitalVideoDisc class.

```
@Override
public int compareTo(Media o) {
    DigitalVideoDisc o1= (DigitalVideoDisc) o;
    int comp= this.getTitle().compareTo(o1.getTitle());
    if(this.getLength()> o1.getLength()) {
        comp= 1;
    }
    if(this.getCost()> o.getCost()) {
        comp= 1;
    }
    return comp;
}
```

Example:

```java
public class TestPassingParameter {

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

        ArrayList<String> authors= new ArrayList<String>();
        authors.add("Volvo");
        authors.add("BMW");
        authors.add("Ford");

        ArrayList<Track> CDs= new ArrayList<Track>();
        Track t1= new Track("Episode 1", 11);
        Track t2= new Track("Episode 2", 2);
        CDs.add(t1);
        CDs.add(t2);

        CompactDisc cd= new CompactDisc(1, "Gaoranger", "Super Sentai", 30.15f, "Kurosaki", "Ichigo", CDs);
        DigitalVideoDisc dvd= new DigitalVideoDisc(2, "Star wars", "Science fiction", 24.95f, 87, "Geogre Lucas");
        Book book1= new Book(3, "Murphy", "Psychology", 17.65f, authors);
        Book book2= new Book(4, "OOP", "IT", 15.55f, authors);

        mediae.add(cd);
        mediae.add(dvd);
        mediae.add(book1);
        mediae.add(book2);

        for(Media m: mediae) {
            System.out.println(m.toString());
            cart.addMedia(m);
        }
        System.out.println(cart.totalCost());
        cart.sortByTitleCost();
    }
}
```

```
CD - [Gaoranger] - [Super Sentai] - [Kurosaki] - [Ichigo] - [13] : [30.15]$
+[Episode 1] - [11 minutes]
+[Episode 2] - [2 minutes]
DVD - [Star wars] - [Science fiction] - [Geogre Lucas] - [87] : [24.95]$
Book - [Murphy] - [Psychology] - [[Volvo, BMW, Ford]] : [17.65]$
Book - [OOP] - [IT] - [[Volvo, BMW, Ford]] : [15.55]$
88.3
Sorted by Title-Cost:
CD - [Gaoranger] - [Super Sentai] - [Kurosaki] - [Ichigo] - [13] : [30.15]$
+[Episode 1] - [11 minutes]
+[Episode 2] - [2 minutes]
Book - [Murphy] - [Psychology] - [[Volvo, BMW, Ford]] : [17.65]$
Book - [OOP] - [IT] - [[Volvo, BMW, Ford]] : [15.55]$
DVD - [Star wars] - [Science fiction] - [Geogre Lucas] - [87] : [24.95]$
```