

Lab 02: Problem Modeling and Encapsulation

4. UML Class Diagram for use cases related to cart management

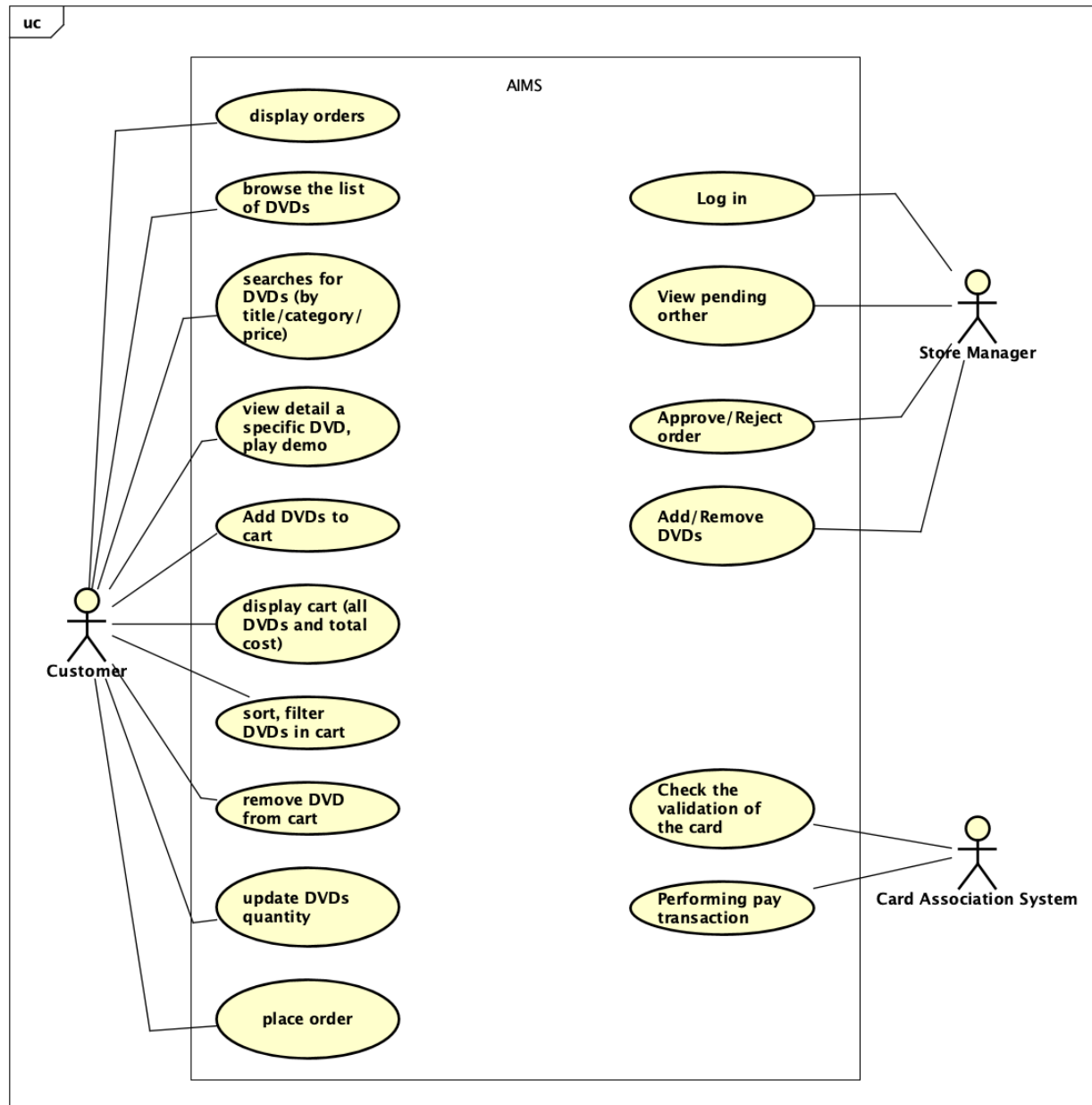


Figure 1 Use case Diagram

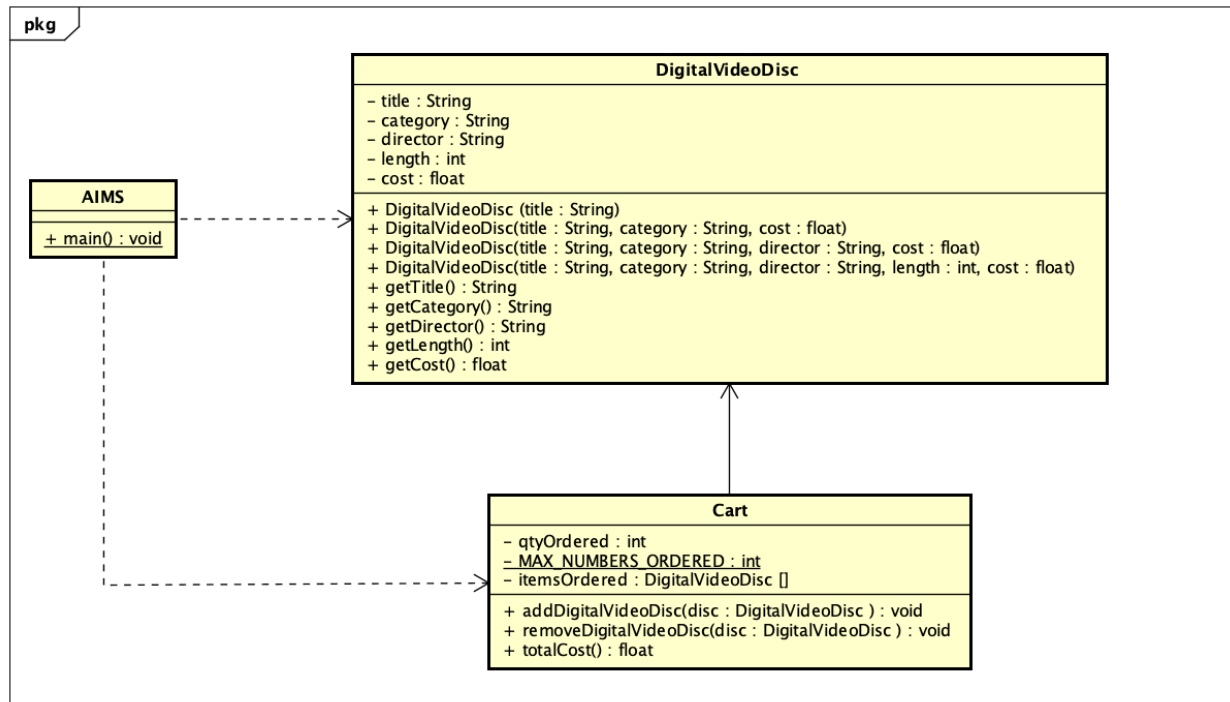


Figure 2 Class Diagram



5. Create Aims Class

week6 > AIMS.java > AIMS > main(String[])

```

1  public class AIMS{
    Run | Debug
2  |
3  | public static void main(String[] args) {
4  | }
5  }
  
```

6. Create the DigitalVideoDisc class and its attributes

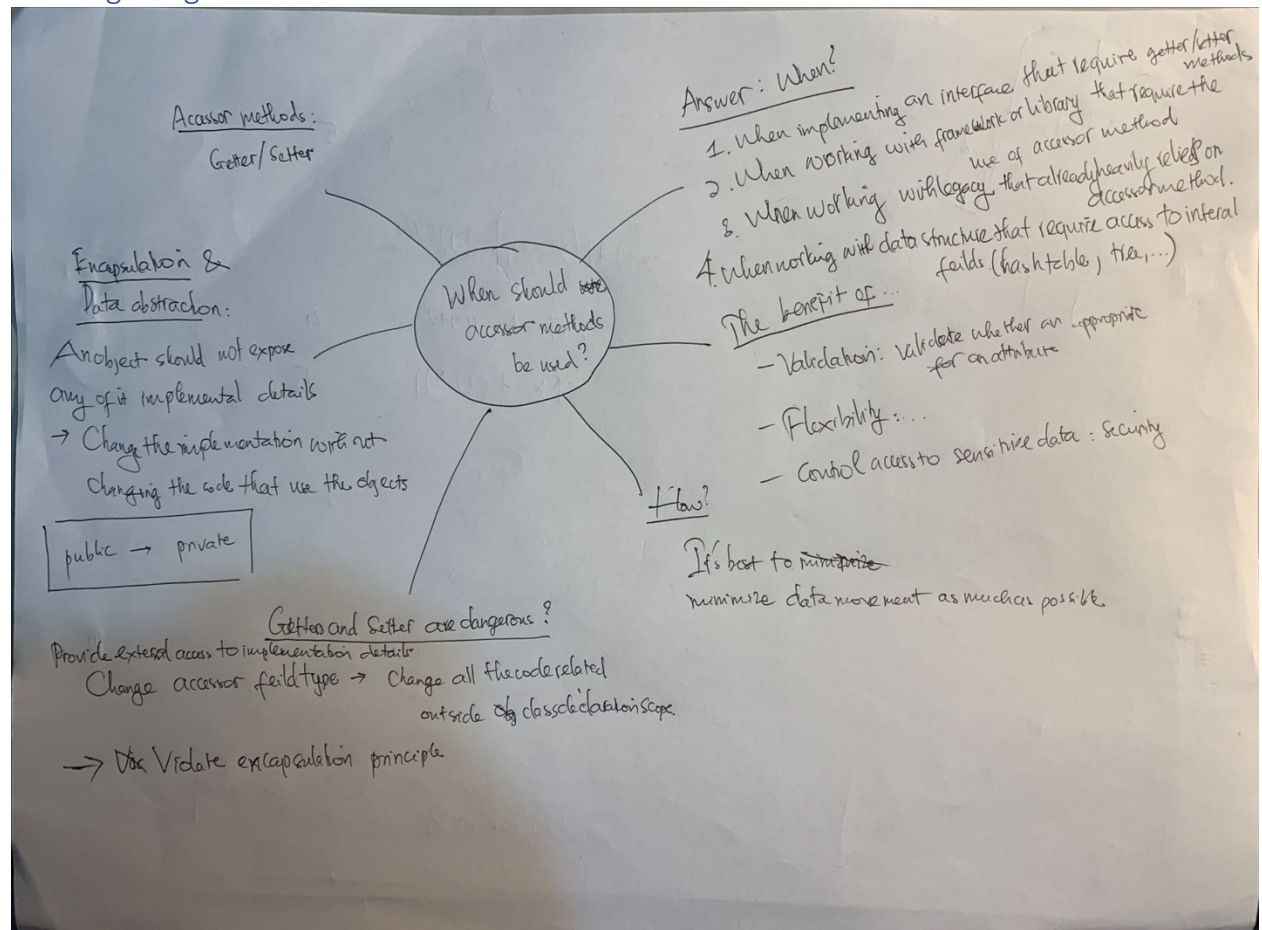
week6 >  DigitalVideoDisc.java >  DigitalVideoDisc

```
1
2  public class DigitalVideoDisc {
3      private String title;
4      private String category;
5      private String director;
6      private int length;
7      private float cost;
8
9
10
11 }
12
```

7. Create accessors and mutators for the class DigitalVideoDisc

```
public class DigitalVideoDisc {  
    private String title;  
    private String category;  
    private String director;  
    private int length;  
    private float cost;  
  
    public String getTitle() {  
        return title;  
    }  
    public String getCategory() {  
        return category;  
    }  
    public String getDirector() {  
        return director;  
    }  
    public int getLength() {  
        return length;  
    }  
    public float getCost() {  
        return cost;  
    }  
}
```

Reading Assignment:



8. Create Constructor method

```
public class DigitalVideoDisc {  
    private String title;  
    private String category;  
    private String director;  
    private int length;  
    private float cost;  
  
    public DigitalVideoDisc(String title) {  
        this.title = title;  
    }  
  
    public DigitalVideoDisc(String title, String category, float cost) {  
        this.title = title;  
        this.category = category;  
        this.cost = cost;  
    }  
  
    public DigitalVideoDisc(String title, String category, String director, float cost) {  
        this.title = title;  
        this.category = category;  
        this.director = director;  
        this.cost = cost;  
    }  
  
    public DigitalVideoDisc(String title, String category, String director, int length, float cost) {  
        this.title = title;  
        this.category = category;  
        this.director = director;  
        this.length = length;  
        this.cost = cost;  
    }  
}
```

Question:

If you create a constructor method to build a DVD by title then create a constructor method to build a DVD by category. Does JAVA allow you to do this?

Answer:

Since the two Constructors have the same signature `DigitalVideoDisc(String)`, then it is not allowed.

```
9      public DigitalVideoDisc(String title) {
10          this.title = title;
11      }
12      public DigitalVideoDisc(String category) {
13          this.category = category;
14      }
15
16      public DigitalVideoDisc(String title, String category, float c
```

PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL Filter (e.g. text, **/...)

✓ DigitalVideoDisc.java week6 2

- ✗ Duplicate method DigitalVideoDisc(String) in type DigitalV... Java(67109219) [Ln 9, Col 12]
- ✗ Duplicate method DigitalVideoDisc(String) in type Digital... Java(67109219) [Ln 12, Col 12]

9. Create the Cart class to work with DigitalVideoDisc

```
week6 > Cart.java > Cart
1  public class Cart {
2      public static final int MAX_NUMBERS_ORDERED = 50;
3      private DigitalVideoDisc[] itemsOrdered = new DigitalVideoDisc[MAX_NUMBERS_ORDERED];
4      private int qtyOrdered;
5      public void addDigitalVideoDisc(DigitalVideoDisc disc) {
6          if (qtyOrdered >= MAX_NUMBERS_ORDERED){
7              System.out.println(x:"The cart is already full, please remove some items be
8              return;
9          }
10         itemsOrdered[qtyOrdered] = disc;
11         ++qtyOrdered;
12         System.out.println(x:"The disc has been added");
13         if (qtyOrdered >= MAX_NUMBERS_ORDERED){
14             System.out.println(x:"The cart is almost full");
15         }
16     }
17 }
```

10. Create Carts of DigitalVideoDiscs

```
week6 > AIMS.java > AIMS
1 public class AIMS {
    Run | Debug
2     public static void main(String[] args) {
3         Cart order = new Cart();
4         DigitalVideoDisc dvd1 = new DigitalVideoDisc(title:"The Lion King", category:"/
5         order.addDigitalVideoDisc(dvd1);
6         DigitalVideoDisc dvd2 = new DigitalVideoDisc(title:"Start Wars", category:"Scie
7         order.addDigitalVideoDisc(dvd2);
8         DigitalVideoDisc dvd3 = new DigitalVideoDisc(title:"Aladin", category:"Animatio
9         order.addDigitalVideoDisc(dvd3);
10
11        System.out.println(order);
12        order.removeDigitalVideoDisc(dvd3);
13
14        System.out.println(order);
15    }
16 }
```

PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL

```
The disc has been added
The disc has been added
The disc has been added
The cart has 3 items
Item #1, Title: The Lion King, Cost: 19.950001
Item #2, Title: Start Wars, Cost: 24.950001
Item #3, Title: Aladin, Cost: 18.990000
Total Cost: 63.889999

The disc has been removed
The cart has 2 items
Item #1, Title: The Lion King, Cost: 19.950001
Item #2, Title: Start Wars, Cost: 24.950001
Total Cost: 44.900002

haidohong@MacBook-Air-cua-haidohong week6 %
```


11. Removing items from the cart

```
16     }
17     public void removeDigitalVideoDisc(DigitalVideoDisc disc) {
18         for (int i = 0; i < qtyOrdered; ++i){
19             if (itemsOrdered[i] == disc) {
20                 --qtyOrdered;
21                 System.out.println(x:"The disc has been removed");
22                 for (int j = i + 1; j < qtyOrdered; ++j){
23                     itemsOrdered[j - 1] = itemsOrdered[j];
24                 }
25                 itemsOrdered[qtyOrdered] = null;
26                 return;
27             }
28         }
29         System.out.println(x:"Cannot find the item.");
30     }
31     public float totalCost(){
```