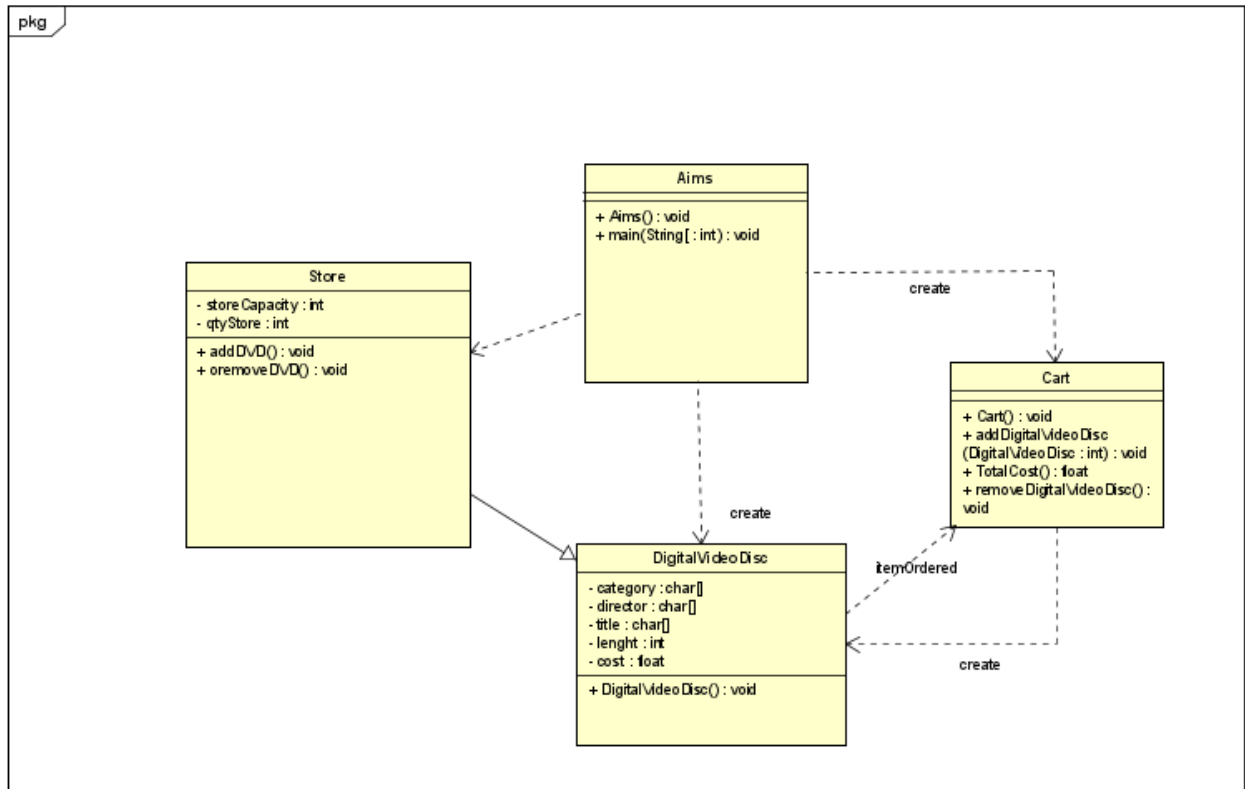


Name: Nguyen Duc Hoang

ID: 20235938

1. Use-case diagram & class diagram:




```

public void addDigitalVideoDisc(DigitalVideoDisc [] dvdList) { no usages
    for (DigitalVideoDisc disc : dvdList) {
        if (qtyOrdered < MAX_NUMBERS_ORDERED) {
            itemsOrdered[qtyOrdered] = disc;
            qtyOrdered += 1;
            System.out.println("DVD list added");
        } else {
            System.out.println("Maximum capacity reached");
        }
    }
}

public void addDigitalVideoDisc(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) { no usages
    if (qtyOrdered + 1 < MAX_NUMBERS_ORDERED) {
        itemsOrdered[qtyOrdered] = dvd1;
        itemsOrdered[qtyOrdered + 1] = dvd2;
        qtyOrdered += 2;
        System.out.println("2 DVD added");
    } else {
        System.out.println("Maximum capacity reached");
    }
}

```

I choose array parameter because it is easy to add DVDs.

3. Passing parameter:

```

1 package hust.soict.ITE7.test.disc;
2
3 import hust.soict.ITE7.aims.disc.DigitalVideoDisc;
4
5 public class TestPassingParameter {
6
7     public static void main(String[] args) {
8         DigitalVideoDisc jungleDVD = new DigitalVideoDisc(title: "Jungle");
9         DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc(title: "Cinderella");
10        swap(jungleDVD, cinderellaDVD);
11        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
12        System.out.println("cinderella dvd title: " + cinderellaDVD.getTitle());
13        changeTitle(jungleDVD, cinderellaDVD.getTitle());
14        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
15    }
16
17    public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) { 1 usage
18        String tempTitle = dvd1.getTitle();
19        dvd1.setTitle(dvd2.getTitle());
20        dvd2.setTitle(tempTitle);
21    }
22
23    public static void changeTitle(DigitalVideoDisc dvd, String title) { 1 usage
24        String oldTitle = dvd.getTitle();
25        dvd.setTitle(title);
26        dvd = new DigitalVideoDisc(oldTitle);
27    }
28 }

```

- Java is a **pass-by-value** language, meaning that when objects are passed to methods, the **value** of the reference (i.e., the memory address) is passed, not the actual object.
- Because it's just a method which is a way to copy, so the original object is not changed.
- The func changeTitle take all address and modify at that, so it will change the original object.

4. Debug:

```

package hust.socit.IEE7.test.disc;
import hust.socit.IEE7.aims.disc.DigitalVideoDisc;

public class TestPassingParameter {

    public static void main(String[] args) {
        DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
        DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc("Cinderella");
        swap(jungleDVD, cinderellaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
        System.out.println("cinderella dvd title: " + cinderellaDVD.getTitle());
        changeTitle(jungleDVD, cinderellaDVD.getTitle());
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
    }

    public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String tempTitle = dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(tempTitle);
    }

    public static void changeTitle(DigitalVideoDisc dvd, String title) {
        String oldTitle = dvd.getTitle();
        dvd.setTitle(title);
        dvd = new DigitalVideoDisc(oldTitle);
    }
}

```

```

"C:\Program Files\Java\jdk-22\bin\java.exe" -agentlib:jwp-transport=dt_socket,address=127.0.0.1:56431,suspend=y,server=n -javaagent:C:\Users\Admin\AppData\Local\JetBrains\IntelliJ\Idea2024.2\captureAgent\debugger-agent.jar
Connected to the target VM, address: '127.0.0.1:56431', transport: 'socket'
jungle dvd title: Cinderella
cinderella dvd title: Jungle
jungle dvd title: Jungle
Disconnected from the target VM, address: '127.0.0.1:56431', transport: 'socket'
Process finished with exit code 0

```

5. Classifier and Instance:

```

1 package hust.soiict.ITE7.aims.disc;
2 public class DigitalVideoDisc { 44 usages
3     private String title; 7 usages
4     private String category; 4 usages
5     private String director; 3 usages
6     private int length; 2 usages
7     private float cost; 4 usages
8     private int id; 6 usages
9     private static int nbDigitalVideoDiscs = 0; 8 usages
10    > public String getTitle() { return title; }
11    > public String getCategory() { return category; }
12    > public String getDirector() { return director; }
13    > public int getLength() { return length; }
14    > public float getCost() { return cost; }
15    > public int getId() { return id; }
16    public DigitalVideoDisc(String title) { 3 usages
17        super();
18        this.title = title;
19        this.id = nbDigitalVideoDiscs;
20        nbDigitalVideoDiscs += 1;
21    }
22    public DigitalVideoDisc(String title, String category, float cost) { 3 usages
23        super();
24        this.title = title;
25        this.category = category;
26        this.cost = cost;
27        this.id = nbDigitalVideoDiscs;
28        nbDigitalVideoDiscs += 1;
29    }
30    public DigitalVideoDisc(String title, String category, String director, float cost) { no usages
31        super();
32        this.title = title;
33        this.category = category;
34        this.director = director;
35        this.cost = cost;
36        this.id = nbDigitalVideoDiscs;
37        nbDigitalVideoDiscs += 1;
38    }
39    public DigitalVideoDisc(String title, String category, String director, int length, float cost) { 6 usages
40        super();
41        this.title = title;
42        this.category = category;
43        this.director = director;
44        this.length = length;
45        this.cost = cost;
46        this.id = nbDigitalVideoDiscs;
47        nbDigitalVideoDiscs += 1;
48    }
49 }

```

6. Open the Cart class:

- String.

```

72 public void print() { 1 usage
73     System.out.println("*****CART*****");
74     System.out.println("Ordered Items:");
75     for (int i = 0; i < qtyOrdered; i++) {
76         System.out.print((i + 1) + ". DVD - ");
77         if (itemsOrdered[i].getTitle() != null) {
78             System.out.print(itemsOrdered[i].getTitle() + " - ");
79         }
80         if (itemsOrdered[i].getCategory() != null) {
81             System.out.print(itemsOrdered[i].getCategory() + " - ");
82         }
83         if (itemsOrdered[i].getDirector() != null) {
84             System.out.print(itemsOrdered[i].getDirector() + " - ");
85         }
86         if (itemsOrdered[i].getLength() != 0) {
87             System.out.print(itemsOrdered[i].getLength() + ": ");
88         }
89         if (itemsOrdered[i].getCost() != 0) {
90             System.out.println(itemsOrdered[i].getCost() + " $");
91         }
92     }
93     System.out.println("Total cost: " + this.totalCost() + " $");
94     System.out.println("*****");
95 }

```

```

97     public void search(int id) { 1 usage
98         boolean found = false;
99         for (int i = 0; i < qtyOrdered; i++) {
100             if (itemsOrdered[i].isMatch(id)) {
101                 System.out.print("Found: " + (i + 1) + ". DVD - ");
102                 if (itemsOrdered[i].getTitle() != null) {
103                     System.out.print(itemsOrdered[i].getTitle() + " - ");
104                 }
105                 if (itemsOrdered[i].getCategory() != null) {
106                     System.out.print(itemsOrdered[i].getCategory() + " - ");
107                 }
108                 if (itemsOrdered[i].getDirector() != null) {
109                     System.out.print(itemsOrdered[i].getDirector() + " - ");
110                 }
111                 if (itemsOrdered[i].getLength() != 0) {
112                     System.out.print(itemsOrdered[i].getLength() + " ");
113                 }
114                 if (itemsOrdered[i].getCost() != 0) {
115                     System.out.println(itemsOrdered[i].getCost() + " $");
116                 }
117                 found = true;
118             }
119         }
120         if (found == false) {
121             System.out.println("No match is found");
122         }
123     }
124 }

```

```

25     public void search(String title) { 1 usage
26         boolean found = false;
27         for (int i = 0; i < qtyOrdered; i++) {
28
29             if (itemsOrdered[i].isMatch(title)) {
30                 System.out.print("Found: " + (i + 1) + ". DVD - ");
31                 if (itemsOrdered[i].getTitle() != null) {
32                     System.out.print(itemsOrdered[i].getTitle() + " - ");
33                 }
34                 if (itemsOrdered[i].getCategory() != null) {
35                     System.out.print(itemsOrdered[i].getCategory() + " - ");
36                 }
37                 if (itemsOrdered[i].getDirector() != null) {
38                     System.out.print(itemsOrdered[i].getDirector() + " - ");
39                 }
40                 if (itemsOrdered[i].getLength() != 0) {
41                     System.out.print(itemsOrdered[i].getLength() + " ");
42                 }
43                 if (itemsOrdered[i].getCost() != 0) {
44                     System.out.println(itemsOrdered[i].getCost() + " $");
45                 }
46                 found = true;
47             }
48         }
49         if (found == false) {
50             System.out.println("No match is found");
51         }
52     }
53 }
54 }

```

```

↑ DVD added
↓ DVD added
DVD added
*****CART*****
Ordered Items:
1. DVD - The Lion King - Animation - Roger Allers - 87: 19.95 $
2. DVD - Star Wars - Science Fiction - George Lucas - 87: 24.95 $
3. DVD - Aladdin - Animation - 18.99 $
Total cost: 63.89 $
*****
Found: 3. DVD - Aladdin - Animation - 18.99 $
Found: 2. DVD - Star Wars - Science Fiction - George Lucas - 87: 24.95 $

Process finished with exit code 0

```

SP_Java-main > Lab03 > @AlmsProject > src > hust > solct > ITE7 > test > cart > @CartTest

22:2 CRLF UTF-8 4 spaces

7. Impliment Store class:

```

1 package hust.soict.I7E7.test.store;
2 import hust.soict.I7E7.aims.disc.DigitalVideoDisc;
3 import hust.soict.I7E7.aims.store.Store;
4
5 public class StoreTest {
6
7     public static void main(String[] args) {
8         Store store = new Store();
9         DigitalVideoDisc dvd1 = new DigitalVideoDisc( title: "The Lion King",
10             category: "Animation", director: "Roger Allers", length: 87, cost: 19.95f);
11         store.addDVD(dvd1);
12         DigitalVideoDisc dvd2 = new DigitalVideoDisc( title: "Star Wars",
13             category: "Science Fiction", director: "George Lucas", length: 87, cost: 24.95f);
14         store.addDVD(dvd2);
15         DigitalVideoDisc dvd3 = new DigitalVideoDisc( title: "Aladdin",
16             category: "Animation", cost: 18.99f);
17         store.addDVD(dvd3);
18         store.removeDVD(dvd2);
19     }
20 }

```

8. String, String Builder & StringBuffer:

```

1 package hust.soict.I7E7.garbage;
2
3 import java.util.Random;
4
5 public class ConcatenationInLoops {
6     public static void main(String[] args) {
7         Random r = new Random( seed: 123);
8         long start = System.currentTimeMillis();
9         String s = "";
10         for (int i = 0; i < 65536; i++) {
11             s += r.nextInt( bound: 2);
12         }
13         System.out.println(System.currentTimeMillis() - start);
14         r = new Random( seed: 123);
15         start = System.currentTimeMillis();
16         StringBuilder sb = new StringBuilder();
17         for (int i = 0; i < 65536; i++) {
18             sb.append(r.nextInt( bound: 2));
19         }
20         s = sb.toString();
21         System.out.println(System.currentTimeMillis() - start);
22     }
23 }

```

```

1 package hust.soict.I7E7.garbage;
2
3 import java.io.IOException;
4 import java.nio.file.Files;
5 import java.nio.file.Paths;
6
7 public class GarbageCreator {
8     public static void main(String[] args) throws IOException {
9         String filepath = "C:/Users/Vo/Desktop/large_input.txt";
10         byte[] inputBytes = {0};
11         long startTime, endTime;
12         inputBytes = Files.readAllBytes(Paths.get(filepath));
13         startTime = System.currentTimeMillis();
14         String outputString = "";
15         for (byte b : inputBytes) {
16             outputString += (char)b;
17         }
18         endTime = System.currentTimeMillis();
19         System.out.println(endTime - startTime);
20     }
21 }
22

```

```
1 package hust.sict.ile7.garbage;
2
3 import java.io.IOException;
4 import java.nio.file.Files;
5 import java.nio.file.Paths;
6
7 public class NoGarbage {
8     public static void main(String[] args) throws IOException {
9         String filepath = "C:/Users/Vo/Desktop/large_input.txt";
10        byte[] inputBytes = {0};
11        long startTime, endTime;
12        inputBytes = Files.readAllBytes(Paths.get(filepath));
13        startTime = System.currentTimeMillis();
14        StringBuilder outputString = new StringBuilder();
15        for (byte b : inputBytes) {
16            outputString.append(b);
17        }
18        endTime = System.currentTimeMillis();
19        System.out.println(endTime - startTime);
20    }
21 }
22
```