

# Báo cáo Thực hành Lab OOP Bài 3

**Họ và tên: Nguyễn Khoa Ninh**

**MSSV: 20226117**

# 1. Branch your repository

```
ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103 (main)
$ git checkout -b release/lab03
Switched to a new branch 'release/lab03'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103 (release/lab03)
$ cd Lab_3
bash: cd: Lab_3: No such file or directory

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103 (release/lab03)
$ cd Lab_0
Lab_01/ Lab_02/ Lab_03/

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103 (release/lab03)
$ cd Lab_0
Lab_01/ Lab_02/ Lab_03/

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103 (release/lab03)
$ cd Lab_0
Lab_01/ Lab_02/ Lab_03/

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103 (release/lab03)
$ cd Lab_03
ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (release/
lab03)
$ |
```

```

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (release/lab03)
$ git checkout -b refactor/apply-release-flow
Switched to a new branch 'refactor/apply-release-flow'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (refactor/apply-release-flow)
$ git checkout -b topic/method-overloading
Switched to a new branch 'topic/method-overloading'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (topic/method-overloading)
$ git checkout -b topic/method-overloading
fatal: a branch named 'topic/method-overloading' already exists

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (topic/method-overloading)
$ git checkout -b topic/passing-parameter
Switched to a new branch 'topic/passing-parameter'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (topic/passing-parameter)
$ git checkout -b topic/class-members
Switched to a new branch 'topic/class-members'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (topic/class-members)
$ git checkout -b feature/print-cart
Switched to a new branch 'feature/print-cart'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (feature/print-cart)
$ git checkout -b feature/search-cart
Switched to a new branch 'feature/search-cart'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (feature/search-cart)
$ git checkout -b topic/store
Switched to a new branch 'topic/store'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (topic/store)
$ git checkout -b refactor/packages
Switched to a new branch 'refactor/packages'

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (refactor/packages)
$

```

```

ADMIN@ASUS-TUF-A15 MINGW64 ~/eclipse-workspace/OOP_Lab_1/IT3103/Lab_03 (refactor/packages)
$ git checkout -b topic/memory-management-string
Switched to a new branch 'topic/memory-management-string'

```

## 2. Working with method overloading

### 2.1 Overloading by differing types of parameter

Phương thức mới đầu tiên addDigitalVideoDisc(DigitalVideoDisc[] dvdList) cho phép thêm một danh sách các đĩa DVD vào giỏ hàng.

```

public int addDigitalVideoDisc(DigitalVideoDisc[] dvdArray) {
    int countAdded = 0;
    for (DigitalVideoDisc disc : dvdArray) {
        if (qtyOrdered == MAX_ITEMS) {
            System.out.println("The cart is full. Can't add more discs.");
            break;
        } else {
            itemsOrdered[qtyOrdered++] = disc;
            System.out.println("The DVD \"" + disc.getTitle() + "\" has been added!");
            countAdded++;
        }
    }
    return countAdded;
}

```

Phương thức mới thứ hai addDigitalVideoDisc(DigitalVideoDisc... dvdArray) sử dụng varargs để cho phép thêm một số lượng tùy ý các đĩa DVD vào giỏ hàng

```

public int addDigitalVideoDisc(DigitalVideoDisc... dvdArray) {
    int countAdded = 0;
    for (DigitalVideoDisc disc : dvdArray) {
        if (qtyOrdered == MAX_ITEMS) {
            System.out.println("The cart is full. Can't add more discs.");
            break;
        } else {
            itemsOrdered[qtyOrdered++] = disc;
            System.out.println("The DVD \"" + disc.getTitle() + "\" has been added!");
            countAdded++;
        }
    }
    return countAdded;
}

```

Cách sử dụng varargs linh hoạt hơn vì nó cho phép thêm bất kỳ số lượng đĩa DVD nào mà không cần chỉ định mảng một cách rõ ràng. Điều này làm thuận tiện hơn khi gọi phương thức. => Chọn varargs

## 2.2. Overloading by differing the number of parameters

```

public int addDigitalVideoDisc(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
    if (qtyOrdered + 2 > MAX_ITEMS) {
        System.out.println("The cart is full or almost full. Can't add more discs.");
        return 0;
    } else {
        itemsOrdered[qtyOrdered++] = dvd1;
        System.out.println("The DVD \"" + dvd1.getTitle() + "\" has been added!");

        itemsOrdered[qtyOrdered++] = dvd2;
        System.out.println("The DVD \"" + dvd2.getTitle() + "\" has been added!");
        return 2;
    }
}

```

## 3. Passing parameter

Trả lời câu hỏi 1: Java là ngôn ngữ lập trình "Pass by Value". Trong Java, khi truyền một tham số cho một phương thức, giá trị của tham số được sao chép và truyền vào phương thức. Điều này có nghĩa là nếu thay đổi giá trị của tham số bên trong phương thức, giá trị của biến gọi phương thức không bị ảnh hưởng.

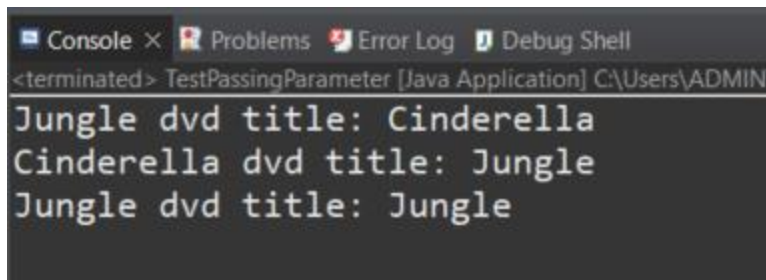
Trả lời câu hỏi 2: Sau khi thực hiện phương thức swap(jungleDVD, cinderellaDVD), tiêu đề của hai đối tượng vẫn giữ nguyên vì trong Java, tham số được truyền vào phương thức là giá trị của đối tượng, không phải là tham chiếu đến đối tượng. Khi ta thay đổi giá trị của tham số bên trong phương thức (như việc đổi chỗ giữa o1 và o2), sự thay đổi này không ảnh hưởng đến giá trị của các đối tượng gốc.

Trả lời câu hỏi 3: Sau khi gọi changeTitle(jungleDVD, cinderellaDVD.getTitle()), tiêu đề của jungleDVD bị thay đổi vì trong phương thức changeTitle, ta thực hiện thay đổi trực tiếp trên đối tượng dvd (được truyền vào phương thức) bằng cách gọi dvd.setTitle(title). Điều này ảnh hưởng trực tiếp đến đối tượng gốc được truyền vào phương thức.

Sửa method swap():

```
1 package hust.soict.dsai.aims.test;
2 import hust.soict.dsai.aims.disc.DigitalVideoDisc;
3
4 public class TestPassingParameter {
5     public static void main(String[] args){
6         DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
7         DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc("Cinderella");
8
9         swap(jungleDVD, cinderellaDVD);
10
11         System.out.println("Jungle dvd title: " + jungleDVD.getTitle());
12         System.out.println("Cinderella dvd title: " + cinderellaDVD.getTitle());
13
14         changeTitle(jungleDVD, cinderellaDVD.getTitle());
15         System.out.println("Jungle dvd title: " + jungleDVD.getTitle());
16     }
17
18     public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
19         DigitalVideoDisc tmp = new DigitalVideoDisc(dvd1.getTitle(), dvd1.getCategory(), dvd1.getDirector(), dvd1.getLength(), dvd1.getCost());
20
21         dvd1.setTitle(dvd2.getTitle());
22         dvd1.setCategory(dvd2.getCategory());
23         dvd1.setDirector(dvd2.getDirector());
24         dvd1.setLength(dvd2.getLength());
25         dvd1.setCost(dvd2.getCost());
26
27         dvd2.setTitle(tmp.getTitle());
28         dvd2.setCategory(tmp.getCategory());
29         dvd2.setDirector(tmp.getDirector());
30         dvd2.setLength(tmp.getLength());
31         dvd2.setCost(tmp.getCost());
32     }
33
34     public static void changeTitle(DigitalVideoDisc dvd, String title){
35         String oldTitle = dvd.getTitle();
36         dvd.setTitle(title);
37         dvd = new DigitalVideoDisc(oldTitle);
38     }
```

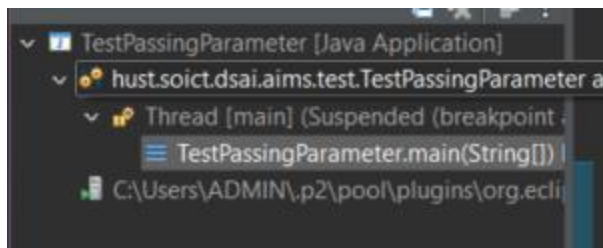
Output:



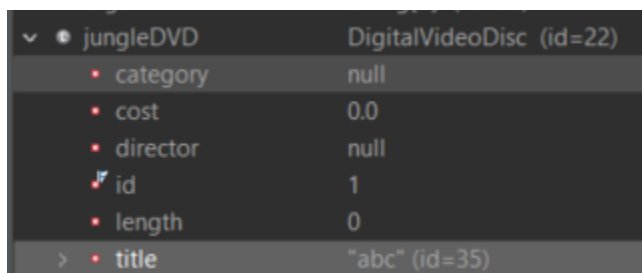
```
<terminated> TestPassingParameter [Java Application] C:\Users\ADMIN
Jungle dvd title: Cinderella
Cinderella dvd title: Jungle
Jungle dvd title: Jungle
```

#### 4. Use debug run

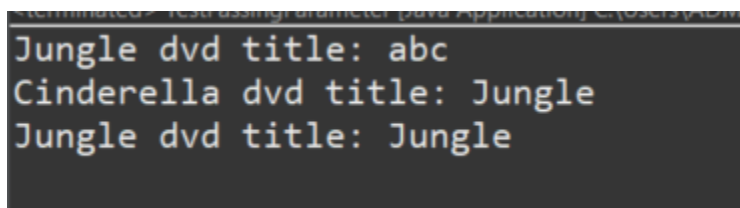
Debug mode:



Thay đổi giá trị title Jungle -> abc trong Variables



▼ jungleDVD	DigitalVideoDisc (id=22)
• category	null
• cost	0.0
• director	null
• id	1
• length	0
> • title	"abc" (id=35)



```
<terminated> TestPassingParameter [Java Application] C:\Users\ADMIN
Jungle dvd title: abc
Cinderella dvd title: Jungle
Jungle dvd title: Jungle
```

#### 5. Classifier Member and Instance Member

```

1 package hust.soict.dsai.aims.disc;
2 public class DigitalVideoDisc {
3
4     private static int nbDigitalVideoDiscs = 0; // Class attribute to track number of discs created
5     private final int id; // Unique ID for each disc
6     private String title;
7     private String category;
8     private String director;
9     private int length;
10    private float cost;
11
12    // Constructor by title
13    public DigitalVideoDisc(String title) {
14        this(title, null, null, 0, 0.0f);
15    }
16
17    // Constructor by category, title, and cost
18    public DigitalVideoDisc(String title, String category, float cost) {
19        this(title, category, null, 0, cost);
20    }
21
22    // Constructor by title, category, director, and cost
23    public DigitalVideoDisc(String title, String category, String director, float cost) {
24        this(title, category, director, 0, cost);
25    }
26
27    // Full constructor with all attributes
28    public DigitalVideoDisc(String title, String category, String director, int length, float cost) {
29        this.title = title;
30        this.category = category;
31        this.director = director;
32        this.length = length;
33        this.cost = cost;
34        this.id = ++nbDigitalVideoDiscs;
35    }
36
37    // Getters
38    public String getTitle() { return title; }
39    public String getCategory() { return category; }
40    public String getDirector() { return director; }
41    public int getLength() { return length; }
42    public float getCost() { return cost; }
43    public int getId() { return id; }
44
45    // Setters with method names reflecting the attribute they modify
46    public void setTitle(String title) { this.title = title; }
47    public void setCategory(String category) { this.category = category; }
48    public void setDirector(String director) { this.director = director; }
49    public void setLength(int length) { this.length = length; }
50    public void setCost(float cost) { this.cost = cost; }
51 }
52

```

6. Open the **Cart** class



```

1 package hust.soict.dsai.aims.cart;
2 import hust.soict.dsai.aims.disc.DigitalVideoDisc;
3 public class Cart {
4     public static final int MAX_ITEMS = 20;
5     private DigitalVideoDisc itemsOrdered[] = new DigitalVideoDisc[MAX_ITEMS];
6     private int qtyOrdered = 0;
7
8     public int addDigitalVideoDisc(DigitalVideoDisc disc) {
9         if (qtyOrdered == MAX_ITEMS) {
10             System.out.println("The cart is full. Can't add more discs.");
11             return 0;
12         } else {
13             itemsOrdered[qtyOrdered++] = disc;
14             System.out.println("The DVD \"" + disc.getTitle() + "\" has been added!");
15             return 1;
16         }
17     }
18
19     public int addDigitalVideoDisc(DigitalVideoDisc... dvdArray) {
20         int countAdded = 0;
21         for (DigitalVideoDisc disc : dvdArray) {
22             if (qtyOrdered == MAX_ITEMS) {
23                 System.out.println("The cart is full. Can't add more discs.");
24                 break;
25             } else {
26                 itemsOrdered[qtyOrdered++] = disc;
27                 System.out.println("The DVD \"" + disc.getTitle() + "\" has been added!");
28                 countAdded++;
29             }
30         }
31         return countAdded;
32     }
33
34     public int addDigitalVideoDisc(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
35         if (qtyOrdered + 2 > MAX_ITEMS) {
36             System.out.println("The cart is full or almost full. Can't add more discs.");
37             return 0;
38         } else {
39             itemsOrdered[qtyOrdered++] = dvd1;
40             System.out.println("The DVD \"" + dvd1.getTitle() + "\" has been added!");
41
42             itemsOrdered[qtyOrdered++] = dvd2;
43             System.out.println("The DVD \"" + dvd2.getTitle() + "\" has been added!");
44             return 2;

```



```

40         System.out.println("The DVD \"" + dvd1.getTitle() + "\" has been added!");
41
42         itemsOrdered[qtyOrdered++] = dvd2;
43         System.out.println("The DVD \"" + dvd2.getTitle() + "\" has been added!");
44         return 2;
45     }
46 }
47
48 public int removeDigitalVideoDisc(DigitalVideoDisc disc) {
49     if (qtyOrdered == 0) {
50         System.out.println("Your cart is empty!");
51         return 0;
52     }
53     for (int i = 0; i < qtyOrdered; i++) {
54         if (itemsOrdered[i].equals(disc)) {
55             for (int j = i; j < qtyOrdered - 1; j++) {
56                 itemsOrdered[j] = itemsOrdered[j + 1];
57             }
58             itemsOrdered[--qtyOrdered] = null;
59             System.out.println("Removed DVD \"" + disc.getTitle() + "\" successfully!");
60             return 1;
61         }
62     }
63     System.out.println("No matching DVD found!");
64     return 0;
65 }
66
67 public float totalCost() {
68     float sum = 0.0f;
69     for (int i = 0; i < qtyOrdered; i++) {
70         if (itemsOrdered[i] != null) {
71             sum += itemsOrdered[i].getCost();
72         }
73     }
74     return sum;
75 }
76
77 public void print() {
78     if (qtyOrdered == 0) {
79         System.out.println("Cart is empty.");
80         return;
81     }
82     StringBuilder output = new StringBuilder("*****CART***** \nOrdered items: \n");
83     for (int i = 0; i < qtyOrdered; i++) {

```

```

76
77 public void print() {
78     if (qtyOrdered == 0) {
79         System.out.println("Cart is empty.");
80         return;
81     }
82     StringBuilder output = new StringBuilder("*****CART***** \nOrdered items: \n");
83     for (int i = 0; i < qtyOrdered; i++) {
84         if (itemsOrdered[i] != null) {
85             output.append(i + 1).append(" ").append(itemsOrdered[i].getTitle()).append(" - [")
86                 .append(itemsOrdered[i].getCategory()).append("] - [")
87                 .append(itemsOrdered[i].getDirector()).append("] - [")
88                 .append(itemsOrdered[i].getLength()).append("]: ")
89                 .append(itemsOrdered[i].getCost()).append(" $\n");
90         }
91     }
92     output.append("Total: ").append(totalCost()).append(" $\n");
93     output.append("*****\n");
94     System.out.println(output);
95 }
96
97 public void searchById(int id) {
98     if (id <= 0 || id > qtyOrdered) {
99         System.out.println("No match found!");
100     } else {
101         DigitalVideoDisc disc = itemsOrdered[id - 1];
102         System.out.println("Result: [" + disc.getTitle() + "] - [" + disc.getCategory() + "] - ["
103             + disc.getDirector() + "] - [" + disc.getLength() + "]: " + disc.getCost() + " $\n");
104     }
105 }
106
107 public void searchByTitle(String title) {
108     for (int i = 0; i < qtyOrdered; i++) {
109         if (itemsOrdered[i].getTitle().equalsIgnoreCase(title)) {
110             System.out.println("Result: [" + itemsOrdered[i].getTitle() + "] - [" + itemsOrdered[i].getCategory()
111                 + "] - [" + itemsOrdered[i].getDirector() + "] - [" + itemsOrdered[i].getLength()
112                 + "]: " + itemsOrdered[i].getCost() + " $\n");
113             return;
114         }
115     }
116     System.out.println("No match found!");
117 }
118 }

```

CartTest class:

The screenshot shows the CartTest class in the left pane and its console output in the right pane. The CartTest class is a public class with a main method that creates a Cart object, adds three DigitalVideoDisc objects, and tests the print, searchById, and searchByTitle methods. The console output shows the results of these tests, including the list of items in the cart and the results of the search methods.

```

1 package hust.soict.dsai.aims.test;
2
3 import hust.soict.dsai.aims.disc.DigitalVideoDisc;
4 import hust.soict.dsai.aims.cart.Cart;
5 public class CartTest {
6     public static void main(String[] args) {
7         Cart cart = new Cart();
8         DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King", "Animation",
9             "Roger Allers", 87, 19.95f);
10        cart.addDigitalVideoDisc(dvd1);
11        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars", "Science Fiction",
12            "George Lucas", 87, 24.95f);
13        cart.addDigitalVideoDisc(dvd2);
14        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladdin", "Animation", 18.99f);
15        cart.addDigitalVideoDisc(dvd3);
16
17        cart.print();
18
19        //Test search by ID method
20        cart.searchById(3);
21        cart.searchById(4);
22
23        //Test search by Title method
24        cart.searchByTitle("The Lion King");
25        cart.searchByTitle("Alan Walker");
26    }
27 }

```

Console Output:

```

The DVD "The Lion King" has been added!
The DVD "Star Wars" has been added!
The DVD "Aladdin" has been added!
*****CART*****
Ordered items:
1. [The Lion King] - [Animation] - [Roger Allers] - [87]: 19.95 $
2. [Star Wars] - [Science Fiction] - [George Lucas] - [87]: 24.95 $
3. [Aladdin] - [Animation] - [null] - [0]: 18.99 $
Total: 63.89 $
*****
Result: [Aladdin] - [Animation] - [null] - [0]: 18.99 $
No match found!
Result: [The Lion King] - [Animation] - [Roger Allers] - [87]: 19.95 $
No match found!

```

## 7. Implement the Store class

Sử dụng LinkedList<>() thay thế cho array thông thường cho *itemsInStore* để tối ưu hóa truy vấn, đơn giản hóa việc thêm và xóa DVD.

```

1 package hust.soict.dsai.aims.store;
2 import java.util.LinkedList;
3
4 import hust.soict.dsai.aims.disc.DigitalVideoDisc;
5
6 public class Store {
7     private LinkedList<DigitalVideoDisc> itemsInStore = new LinkedList<>();
8
9     public boolean checkDVD(DigitalVideoDisc disc) {
10         return itemsInStore.contains(disc);
11     }
12
13     public void removeDVD(DigitalVideoDisc disc) {
14         if (checkDVD(disc)) {
15             itemsInStore.remove(disc);
16             System.out.printf("The DVD \"%s\" has been removed from the store!\n", disc.getTitle());
17         } else {
18             System.out.printf("The DVD \"%s\" is not found in the store!\n", disc.getTitle());
19         }
20     }
21
22     public void addDVD(DigitalVideoDisc disc) {
23         if (!checkDVD(disc)) {
24             itemsInStore.add(disc);
25             System.out.printf("The DVD \"%s\" has been added to the store!\n", disc.getTitle());
26         } else {
27             System.out.printf("The DVD \"%s\" already exists in the store!\n", disc.getTitle());
28         }
29     }
30
31     @Override
32     public String toString() {
33         StringBuilder output = new StringBuilder("STORE\nItems in the store:\n");
34         if (itemsInStore.isEmpty()) {
35             output.append("There are no DVDs in the store!\n");
36         } else {
37             for (DigitalVideoDisc dvd : itemsInStore) {
38                 output.append(String.format("%s - %.2f $\n", dvd.getTitle(), dvd.getCost()));
39             }
40         }
41         return output.toString();
42     }
43 }
44

```

## TestStore class

```

1 package hust.soict.dsai.aims.test;
2
3 import hust.soict.dsai.aims.disc.DigitalVideoDisc;
4 import hust.soict.dsai.aims.store.Store;
5
6 public class TestStore {
7     public static void main(String[] args) {
8         Store store = new Store();
9         DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King", "Animation",
10             "Roger Allers", 87, 19.95f);
11         DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star wars", "Science Fiction",
12             "Geogre Lucas", 87, 24.95f);
13         DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin", "Animation", 18.99f);
14         store.addDVD(dvd1);
15         store.addDVD(dvd2);
16         store.addDVD(dvd3);
17
18         System.out.println(store.toString());
19
20         store.removeDVD(dvd1);
21         store.removeDVD(dvd2);
22         store.removeDVD(dvd3);
23
24         System.out.println(store.toString());
25     }
26 }

```

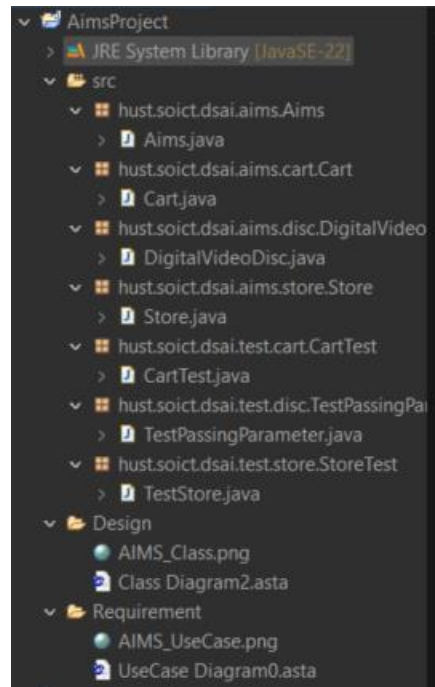
```

The DVD "The Lion King" has been added to the store!
The DVD "Star wars" has been added to the store!
The DVD "Aladin" has been added to the store!
*****STORE*****
Items in the store:
The Lion King - 19.95 $
Star wars - 24.95 $
Aladin - 18.99 $
*****

The DVD "The Lion King" has been removed from the store!
The DVD "Star wars" has been removed from the store!
The DVD "Aladin" has been removed from the store!
*****STORE*****
Items in the store:
There are no DVDs in the store!
*****

```

## 8. Re-organize your projects



## 9. String, StringBuilder and StringBuffer

### ConcatenationInLoops

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

### GarbageCreator

```

1 package hust.soict.dsai.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 filename = "C:\\Users\\ADMIN\\eclipse-workspace\\OtherProjects\\src\\hust\\soict\\dsai\\garbage\\test.txt";
10        byte[] inputBytes = { 0 };
11        long startTime, endTime;
12        inputBytes = Files.readAllBytes(Paths.get(filename));
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 }

```

## NoGarbage

```

1 package hust.soict.dsai.garbage;
2 import java.io.IOException;
3 import java.nio.file.Files;
4 import java.nio.file.Paths;
5
6 public class NoGarbage {
7     public static void main(String[] args) throws IOException {
8         String filename = "C:\\Users\\ADMIN\\eclipse-workspace\\OtherProjects\\src\\hust\\soict\\dsai\\garbage\\test.txt";
9         byte[] inputBytes = { 0 };
10        long startTime, endTime;
11
12        inputBytes = Files.readAllBytes(Paths.get(filename));
13        startTime = System.currentTimeMillis();
14        StringBuilder outputStringBuilder = new StringBuilder("");
15        for(byte b : inputBytes) {
16            outputStringBuilder.append((char)b);
17        }
18        endTime = System.currentTimeMillis();
19        System.out.println(endTime - startTime);
20    }
21 }

```

## 10. Release flow demonstration

Check github

## 11. Cập nhật Diagram

