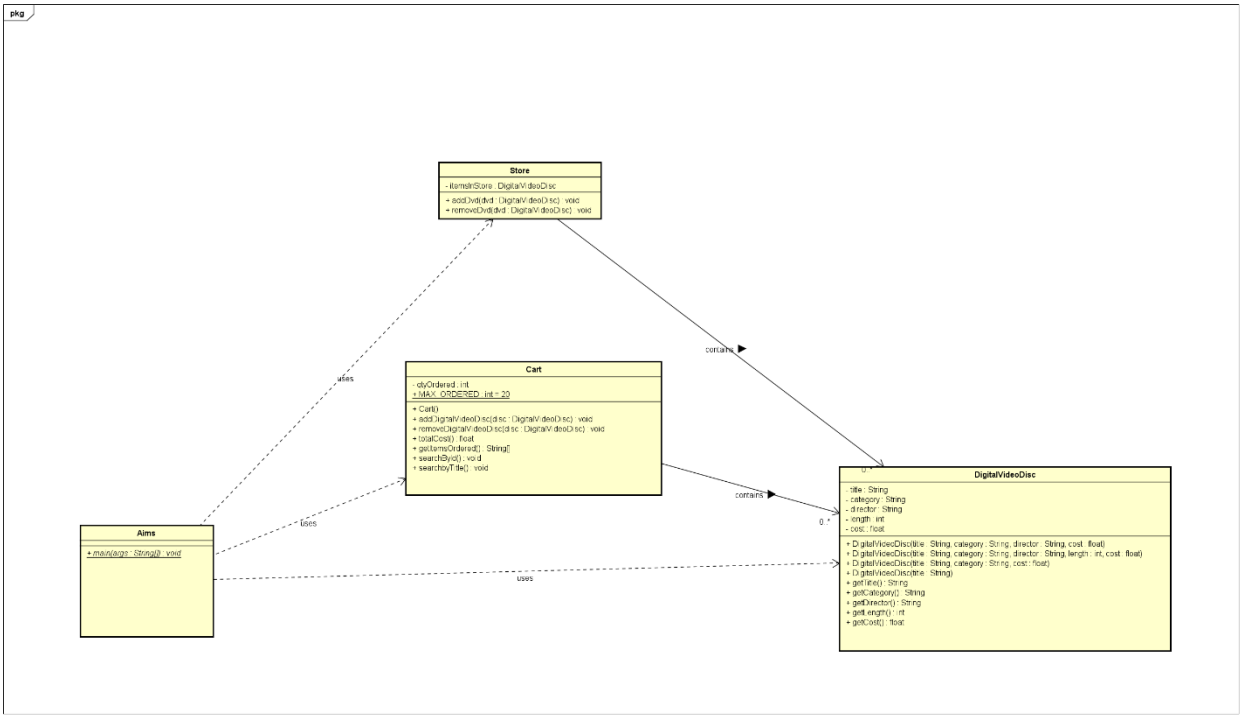
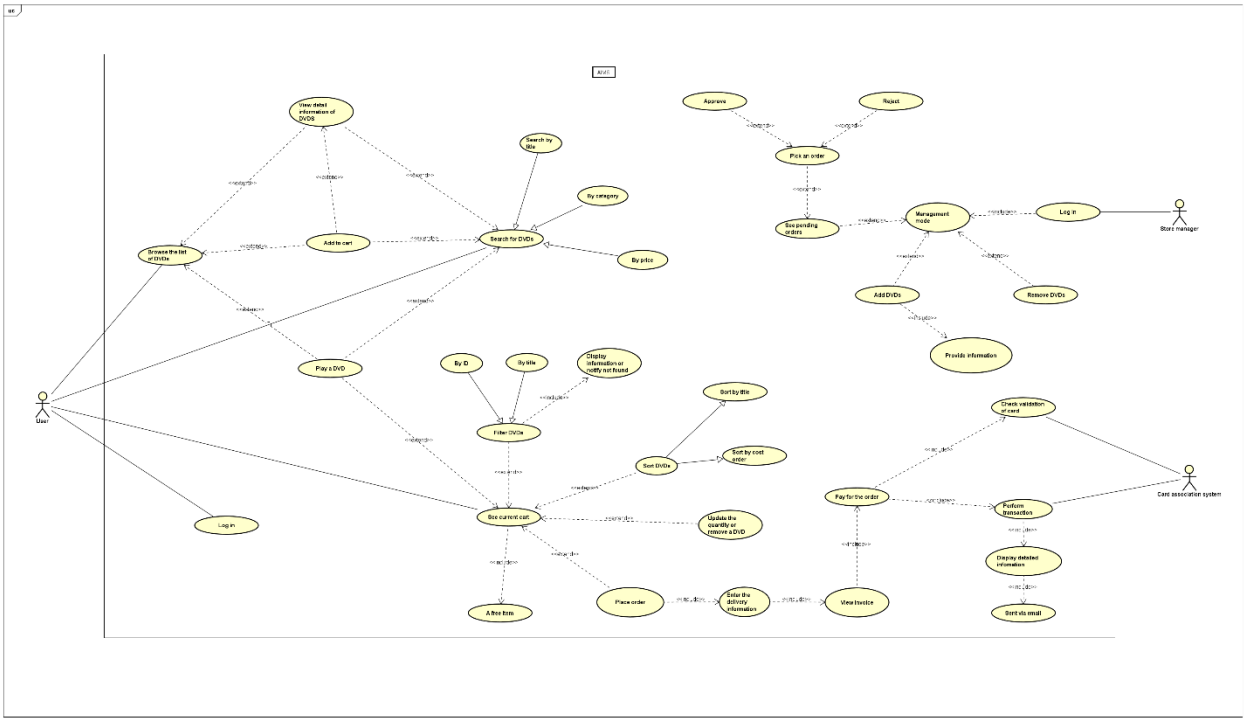
OOP Lab 003 Report

I. Updated Class Diagram and Use Case Diagram



Updated Class Diagram



Updated Use Case Diagram

II. Working with method overloading

Question: *Try to add a method* ***addDigitalVideoDisc*** *which allows to pass an arbitrary number of arguments for* ***dvd****. Compare to an array parameter. What do you prefer in this case?*

```java
//Overloading addDigitalVideoDisc allow listing DVDs as parameters
public void addDigitalVideoDisc( DigitalVideoDisc [] dvdList ) {
    if ( qtyOrdered  >= MAX_ORDERED ) {
        System.out.println(x:"The cart cannot hold more dvds");
    } else {
        for (int i = 0; i < dvdList.length; i++) {
            itemsOrdered[qtyOrdered] =  dvdList[i];
            qtyOrdered += 1;
        }

        System.out.println(x:"Added");
    }
}
//Overloading addDigitalVideoDisc allow adding 2 DVDs
public void addDigitalVideoDisc(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
    if (qtyOrdered + 2 > MAX_ORDERED) {
        System.out.println(x:"The cart cannot hold 2 more DVDs");
        return;
    }
    addDigitalVideoDisc(dvd1);
    addDigitalVideoDisc(dvd2);
}
```

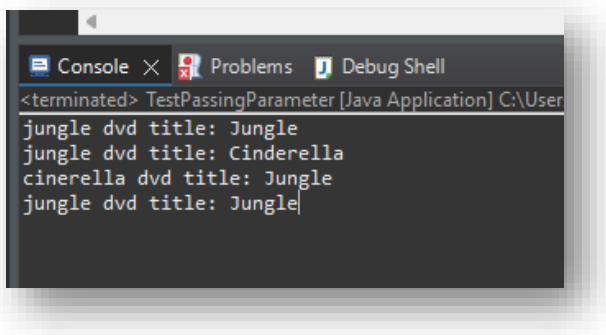
With the array parameter, you can just simply check the length of the ***dvdList*** array to ensure the total number of DVDs in the cart does not exceed the maximum. This provides a more straightforward way to enforce the cart's capacity limit.

III. Passing parameter

Is JAVA a Pass by Value or a Pass by Reference programming language?

- Java is a pass-by-value programming language. For example, if you pass an object into a method in Java (***swap(DVD dvd1, DVD dvd2)***, the method only receives the address values that point to the dvd1 and dvd2 objects in memory. So, if you try to swap the objects by doing ***tmp = dvd1; dvd1 = dvd2; dvd2 = tmp***, it won't work

⇨ This is because the method is only changing the values of the local variables **dvd1** and **dvd2**, which does not affect the original objects.



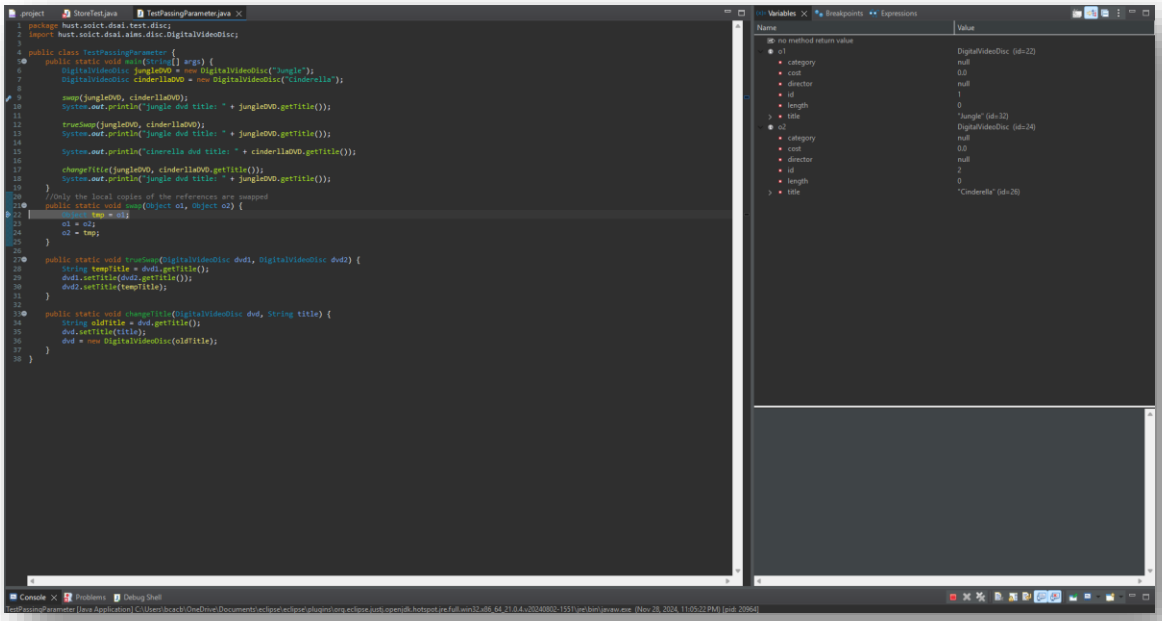
After the call of ***swap(jungleDVD, cinderellaDVD)*** , why does the title of these two objects remain?

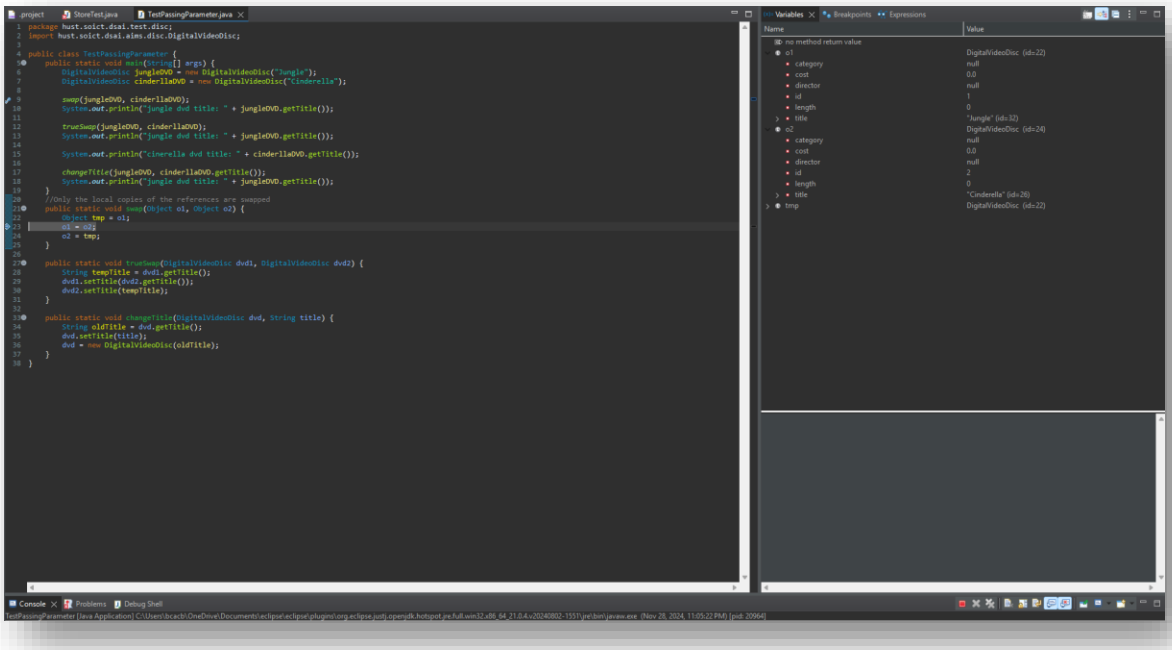
- After calling ***swap(jungleDVD, cinderellaDVD)***, the titles of these two objects remain the same. As mentioned earlier, the ***swap()*** method is only manipulating the local variables, not the original ***jungleDVD*** and ***cinderellaDVD*** objects.

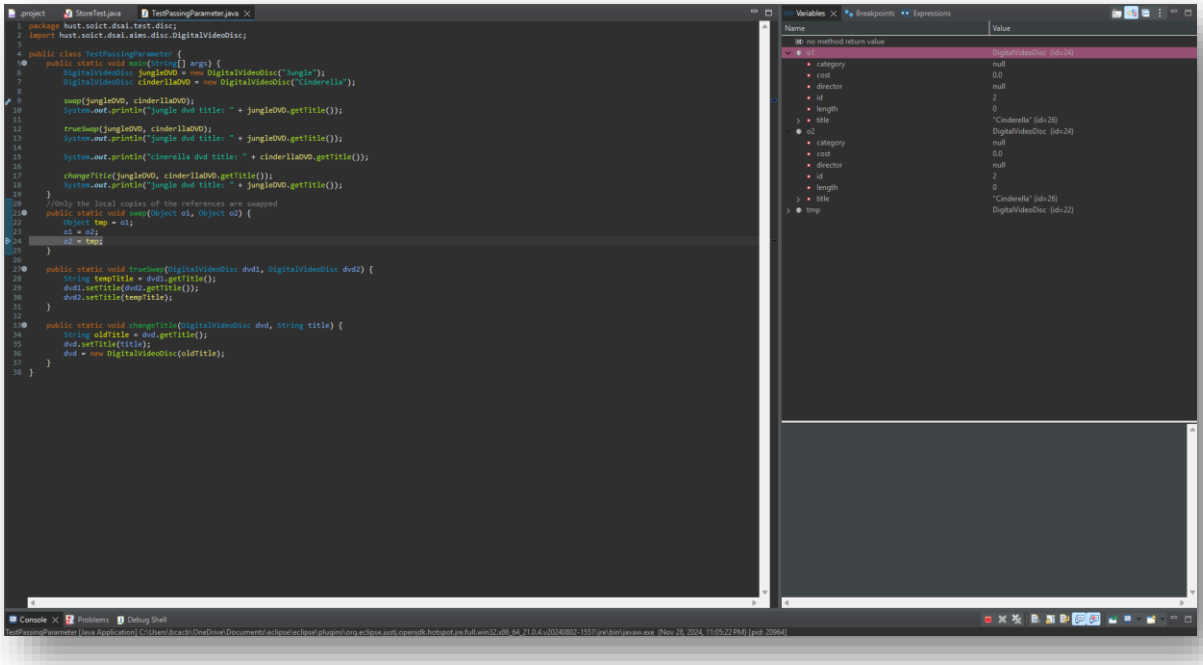
After the call of ***changeTitle(jungleDVD, cinderellaDVD.getTitle())*** why is the title of the JungleDVD changed?

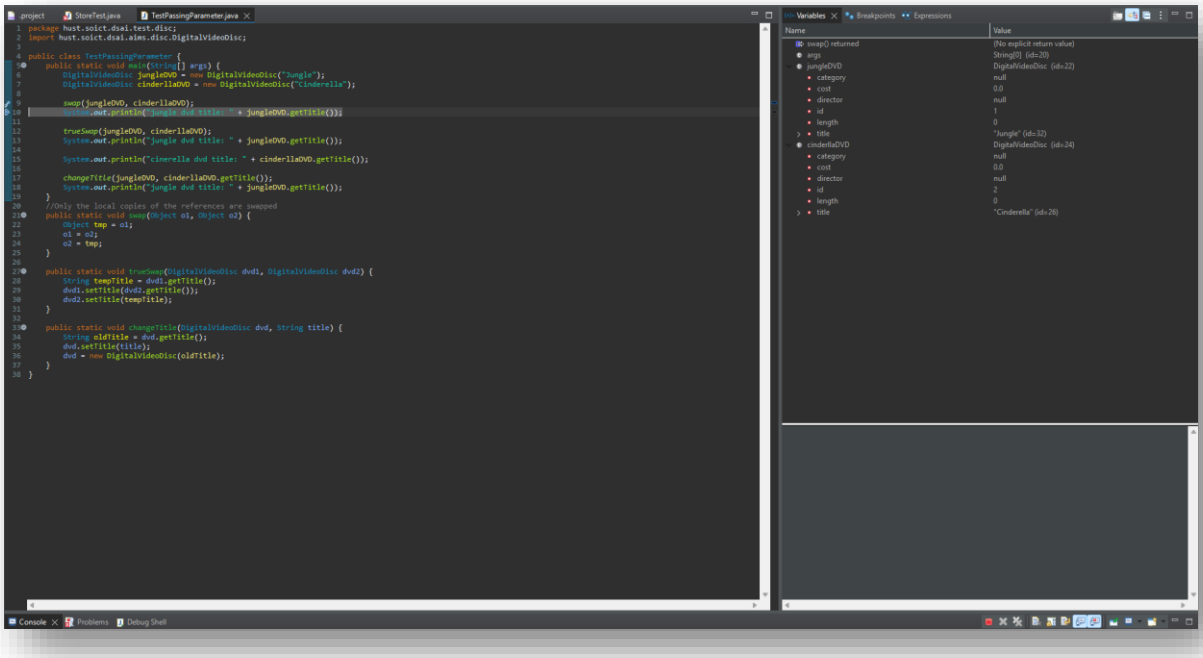
- After calling **changeTitle(jungleDVD, cinderellaDVD.getTitle()),** the title of the **jungleDVD** object is changed. In this case, the **changeTitle()** method is passed a reference to the **jungleDVD** object, so when the title of the **dvd** object (which is the same as **jungleDVD**) is modified, it changes the title of the original **jungleDVD** object, since they both reference the same object in memory.

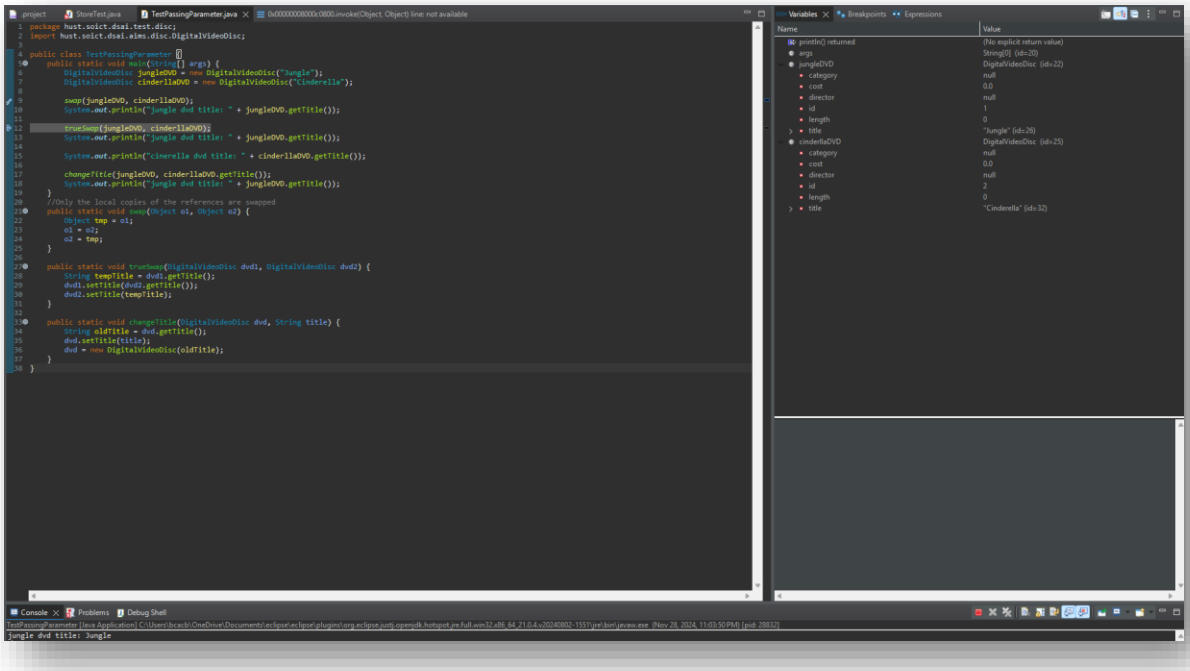
IV. Debugging Java in Eclipse






.project
StoreTest.java
TestPassingParameter.java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;
public class TestPassingParameter {
public static void main(String[] args) {
DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");
swap(jungleDVD, cinderllaDVD);
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
trueSwap(jungleDVD, cinderllaDVD);
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());
changeTitle(jungleDVD, cinderllaDVD.getTitle());
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
}
//Only the local copies of the references are swapped
public static void swap(Object o1, Object o2) {
Object tmp = o1;
o1 = o2;
o2 = tmp;
}
public static void trueSwap(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);
}
}
Variables
Breakpoints
Expressions
Name
Value
no method return value
o1
DigitalVideoDisc (id=24)
category
null
cost
0.0
director
null
id
2
length
0
title
"Cinderella" (id=26)
o2
DigitalVideoDisc (id=24)
category
null
cost
0.0
director
null
id
2
length
0
title
"Cinderella" (id=26)
tmp
DigitalVideoDisc (id=22)
Console
Problems
Debug Shell
TestPassingParameter [Java Application] C:\Users\bcach\OneDrive\Documents\eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.4.v20240802-1551\jre\bin\javaw.exe (Nov 28, 2024, 11:05:22 PM) [pid: 20964]


.project
StoreTest.java
TestPassingParameter.java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;
public class TestPassingParameter {
public static void main(String[] args) {
DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");
swap(jungleDVD, cinderllaDVD);
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
trueSwap(jungleDVD, cinderllaDVD);
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());
changeTitle(jungleDVD, cinderllaDVD.getTitle());
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
}
//Only the local copies of the references are swapped
public static void swap(Object o1, Object o2) {
Object tmp = o1;
o1 = o2;
o2 = tmp;
}
public static void trueSwap(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);
}
}
Variables
Breakpoints
Expressions
Name
Value
swap() returned
(No explicit return value)
args
String[0] (id=20)
jungleDVD
DigitalVideoDisc (id=22)
category
null
cost
0.0
director
null
id
1
length
0
title
"Jungle" (id=32)
cinderllaDVD
DigitalVideoDisc (id=24)
category
null
cost
0.0
director
null
id
2
length
0
title
"Cinderella" (id=26)
Console
Problems
Debug Shell



V. Classifier Member and Instance Member

```java
public DigitalVideoDisc(String title) {
    super();
    this.title = title;
    nbDigitalVideoDiscs += 1;
    this.id = nbDigitalVideoDiscs;
}

public DigitalVideoDisc(String title, String category, float cost) {
    super();
    this.title = title;
    this.category = category;
    this.cost = cost;
    nbDigitalVideoDiscs += 1;
    this.id = nbDigitalVideoDiscs;
}

public DigitalVideoDisc(String title, String category, String director, float cost) {
    super();
    this.title = title;
    this.category = category;
    this.director = director;
    this.cost = cost;
    nbDigitalVideoDiscs += 1;
    this.id = nbDigitalVideoDiscs;
}

public DigitalVideoDisc(String title, String category, String director, int length, float cost) {
    super();
    this.title = title;
    this.category = category;
    this.director = director;
    this.length = length;
    this.cost = cost;
    nbDigitalVideoDiscs += 1;
    this.id = nbDigitalVideoDiscs;
}
```

VI. Open the Cart Class

Write a ***toString()*** method for the ***DigitalVideoDisc*** class. What should be the return type of this method?

```java
public String toString() {

    return "DVD" + "-" + this.title + "-" + this.category + "-" + this.director + "-" + String.valueOf(this.length) + ": " + String.valueOf(this.cost) + "$";
}
```

⇨ The method should return a String

```java
public void printOrders() {
    if (qtyOrdered == 0) {
        System.out.println(x:"The cart is empty");
        return;
    }
    System.out.println(x:"**********************CART**********************");
    System.out.println(x:"Ordered Items:");
    for (int i = 0; i < qtyOrdered; i++) {
        System.out.printf(format:"%d. %s%n", (i + 1), itemsOrdered[i].toString());
    }
    System.out.printf(format:"Total cost: %.2f $%n", totalCost());
    System.out.println(x:"**************************************************");
}

public void searchById(int id) {
    boolean found = false;
    for (int i = 0; i < qtyOrdered; i++) {
        if (itemsOrdered[i].getId() == id) {
            System.out.println("DVD found: " + itemsOrdered[i].toString());
            found = true;
            break;
        }
    }
    if (!found) {
        System.out.println("No DVD found with ID: " + id);
    }
}

public void searchByTitle(String title) {
    if (title == null || title.trim().isEmpty()) {
        System.out.println(x:"Invalid title search");
        return;
    }
    boolean found = false;
    for (int i = 0; i < qtyOrdered; i++) {
        if (itemsOrdered[i].isMatch(title)) {
            System.out.println("DVD found: " + itemsOrdered[i].toString());
            found = true;
            break;
        }
    }
    if (!found) {
        System.out.println("No DVD found with title: " + title);
    }
}
```

.project | StoreTest.java | TestPassingParameter.java | DigitalVideoDisc.java | CartTest.java

```java
package hust.soict.dsai.test.cart;
import hust.soict.dsai.aims.cart.Cart;

public class CartTest {
    public static void main(String[] args) {
        //Create a new cart
        Cart cart = new Cart();

        //Create new dvd objects and add them to the cart
        DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King", "Animation", "Roger Allers", 87, 19.95f);
        cart.addDigitalVideoDisc(dvd1);
        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars", "Science Fiction", "George Lucas", 87, 24.95f);
        cart.addDigitalVideoDisc(dvd2);
        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin", "Animation", 18.99f);
        cart.addDigitalVideoDisc(dvd3);
        //Test the print method
        cart.printOrders();
        //Test the search method
        cart.searchByTitle("Aladin");
        cart.searchByTitle("Ball");
        cart.searchById(1);
        cart.searchById(10);
    }
}
```

Console | Problems | Debug Shell

```
<terminated> CartTest [Java Application] C:\Users\bcacb\OneDrive\Documents\eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0
Added DVD: The Lion King
Added DVD: Star Wars
Added DVD: Aladin
**********************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-Aladin-Animation-null-0: 18.99$
Total cost: 63.89 $
*****************************************************
DVD found: DVD-Aladin-Animation-null-0: 18.99$
No DVD found with title: Ball
DVD found: DVD-The Lion King-Animation-Roger Allers-87: 19.95$
No DVD found with ID: 10
```

VII. Implement the Store Class

```java
package hust.soict.dsai.aims.store;

import hust.soict.dsai.aims.disc.DigitalVideoDisc;
import java.util.ArrayList;

public class Store {
    private ArrayList<DigitalVideoDisc> itemsInStore;

    public Store() {
        itemsInStore = new ArrayList<>();
    }

    public void addDvd(DigitalVideoDisc dvd) {
        if (dvd == null) {
            System.out.println("Cannot add null DVD.");
            return;
        }
        itemsInStore.add(dvd);
        System.out.println("DVD added: " + dvd.getTitle());
    }

    public void removeDvd(DigitalVideoDisc dvd) {
        if (dvd == null) {
            System.out.println("Cannot remove null DVD.");
            return;
        }

        if (itemsInStore.remove(dvd)) {
            System.out.println("DVD removed: " + dvd.getTitle());
        } else {
            System.out.println("DVD not found in the store: " + dvd.getTitle());
        }
    }
}
```

Testing:

```java
package hust.soict.dsai.test.store;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class StoreTest {
    public static void main(String[] args) {
        // Create a new store
        Store store = new Store();

        //Adding DVDs
        DigitalVideoDisc dvd1 = new DigitalVideoDisc("Inception", "Sci-fi", "Christopher Nolan", 148, 19.99f);
        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Interstellar", "Sci-fi", "Christopher Nolan", 169, 24.99f);
        DigitalVideoDisc dvd3 = new DigitalVideoDisc("The Matrix", "Sci-fi", "Wachowski", 136, 15.99f);
        DigitalVideoDisc dvd4 = new DigitalVideoDisc("Inception"); // Duplicate title test

        // Add DVDs to store
        store.addDvd(dvd1);
        store.addDvd(dvd2);
        store.addDvd(dvd3);

        // Test adding null DVD
        store.addDvd(null);

        // Test removing DVDs
        System.out.println("\nTesting removeDVD method:");
        store.removeDvd(dvd2);
        store.removeDvd(dvd3);
    }
}
```

Console | Problems | Debug Shell

```
<terminated> StoreTest [Java Application] C:\Users\bcacb\OneDrive\Documents\eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.4.v2
DVD added: Inception
DVD added: Interstellar
DVD added: The Matrix
Cannot add null DVD.

Testing removeDVD method:
DVD removed: Interstellar
DVD removed: The Matrix
```

String, StringBuilder and StringBuffer

OtherProjects > src > hust > soict > dsai > garbage > GarbageCreator.java > GarbageCreator > main(String[])

```java
package hust.soict.dsai.garbage;

import java.io.*;
import java.nio.file.*;

public class GarbageCreator {
    Run | Debug
    public static void main(String[] args) {
        String filePath = "fat_file.txt"; // Sanitize

        try {
            // Large file for testing
            if (!Files.exists(Paths.get(filePath))) {
                createLargeFile(filePath);
            }

            System.out.println(x:"Starting to read file with String concatenation...");
            long startTime = System.currentTimeMillis();

            // Read by character with String concatenation
            try (FileReader fr = new FileReader(filePath)) {
                String content = "";
                int character;
                int charCount = 0;

                while ((character = fr.read()) != -1) {
                    content += (char) character;
                    charCount++;

                    if (charCount % 1000000 == 0) {
                        System.out.println("Read " + charCount + " characters...");
                    }
                }

                System.out.println("Final string length: " + content.length());
            }

            long endTime = System.currentTimeMillis();
            System.out.println("Time taken: " + (endTime - startTime) + " ms");

        } catch (OutOfMemoryError e) {
            System.out.println(x:"Out of Memory Error occurred!");
            System.out.println("Error: " + e.getMessage());
            e.printStackTrace();
        } catch (IOException e) {
```

PROBLEMS 3 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

```
PS C:\Users\bcacb\OneDrive\Documents\OOP_Labs\OOP-Labs>  & 'C:\Program Files\Java\jdk1.8.0_241\bin\java.exe' '-cp' 'C:\
Starting to read file with String concatenation...
PS C:\Users\bcacb\OneDrive\Documents\OOP_Labs\OOP-Labs>  c:; cd 'c:\Users\bcacb\OneDrive\Documents\OOP_Labs\OOP-Labs';
project\bin' 'hust.soict.dsai.garbage.GarbageCreator'
Starting to read file with String concatenation...
PS C:\Users\bcacb\OneDrive\Documents\OOP_Labs\OOP-Labs>  c:; cd 'c:\Users\bcacb\OneDrive\Documents\OOP_Labs\OOP-Labs';
project\bin' 'hust.soict.dsai.garbage.GarbageCreator'
Starting to read file with String concatenation...
Read 1000000 characters...
```

String concatenation very slow when handle a really big file

OtherProjects > src > hust > soict > dsai > garbage > GarbageCreator.java > GarbageCreator > main(String[])

```java
package hust.soict.dsai.garbage;
import java.io.*;
import java.nio.file.*;

public class GarbageCreator {
    Run | Debug
    public static void main(String[] args) {
        String filePath = "large_file.txt"; // Sanitize
        try {
            // Large file for testing
            if (!Files.exists(Paths.get(filePath))) {
                createLargeFile(filePath);
            }
            System.out.println(x:"Starting to read file with StringBuffer...");
            long startTime = System.currentTimeMillis();
            // Read by character with StringBuffer
            try (FileReader fr = new FileReader(filePath)) {
                StringBuffer content = new StringBuffer();
                int character;
                int charCount = 0;
                while ((character = fr.read()) != -1) {
                    content.append((char) character);
                    charCount++;
                    if (charCount % 1000000 == 0) {
                        System.out.println("Read " + charCount + " characters...");
                    }
                }
                System.out.println("Final string length: " + content.length());
            }
            long endTime = System.currentTimeMillis();
            System.out.println("Time taken: " + (endTime - startTime) + " ms");
        } catch (OutOfMemoryError e) {
            System.out.println(x:"Out of Memory Error occurred!");
            System.out.println("Error: " + e.getMessage());
            e.printStackTrace();
        } catch (IOException e) {
            System.out.println(x:"IO Error occurred!");
            System.out.println("Error: " + e.getMessage());
            e.printStackTrace();
        }
    }

    private static void createLargeFile(String filePath) throws IOException {
        try (FileWriter writer = new FileWriter(filePath)) {
            for (int i = 0; i < 100000000; i++) {
```

PROBLEMS 3 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

```
Read 671000000 characters...
Read 672000000 characters...
Read 673000000 characters...
Read 674000000 characters...
Read 675000000 characters...
Read 676000000 characters...
Read 677000000 characters...
Read 678000000 characters...
Read 679000000 characters...
Read 680000000 characters...
Read 681000000 characters...
Read 682000000 characters...
Read 683000000 characters...
Read 684000000 characters...
Read 685000000 characters...
Read 686000000 characters...
Read 687000000 characters...
Read 688000000 characters...
Read 689000000 characters...
```

Much faster and more efficient with StringBuffer

The main difference between StringBuffer and StringBuilder in Java is that StringBuffer is thread-safe, while StringBuilder is not.

StringBuffer is synchronized, which means its methods can be safely called from multiple threads without the need for external synchronization. This ensures that the operations on the string buffer are atomic and thread safe.