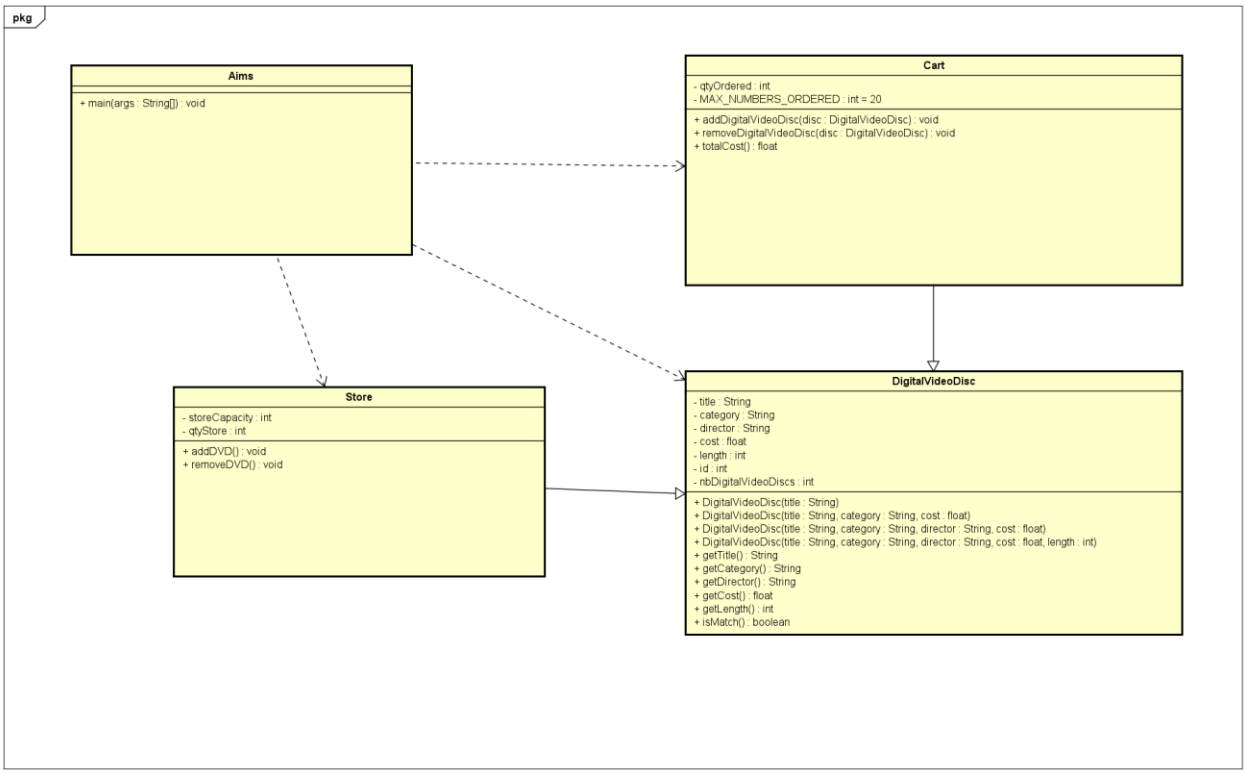
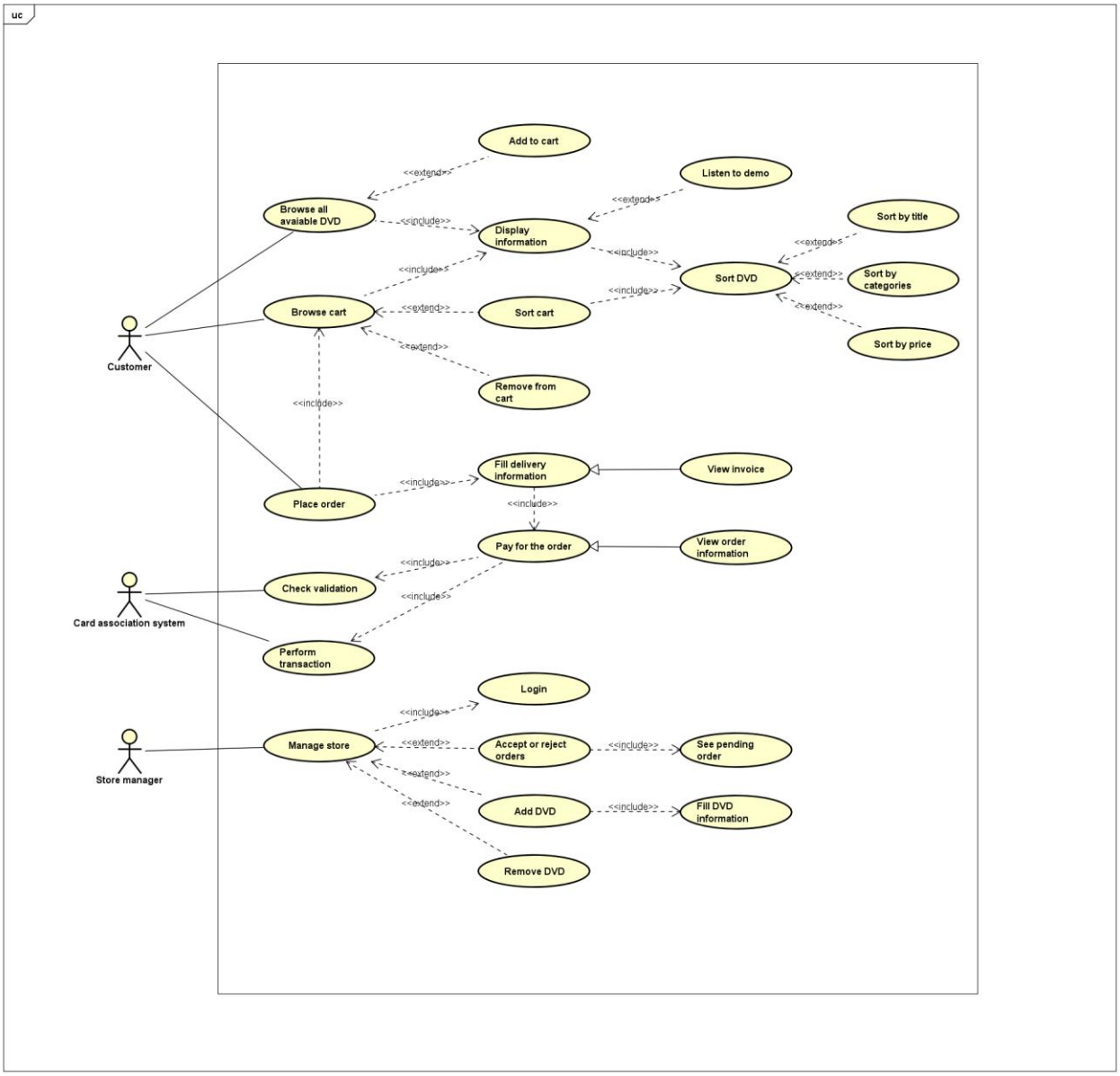
Name: Phạm Chí Bằng

Student ID: 20235477

Use-case diagram and class diagram





Working with method overloading

```
public void addDigitalVideoDisc(DigitalVideoDisc [] dvdList) {
    if  (qtyOrdered >= MAX_NUMBERS_ORDERED) {
        System.out.print("The cart is full");
    }
    else if (qtyOrdered + dvdList.length >= MAX_NUMBERS_ORDERED) {
        System.out.print("the quantity of your adding is too large");

    }
    else {
        for ( int j = 0; j < dvdList.length  ; j++ ) {
            itemOrdered[j + qtyOrdered] = dvdList[j];
        qtyOrdered += dvdList.length;
        }
        System.out.print("Add List of dvds sucessfully");
    }
}

public void addDigitalVideoDisc(DigitalVideoDisc dvd1,DigitalVideoDisc dvd2) {
    if (dvd1.getTitle() == dvd2.getTitle()) {
        System.out.print("Cannot add the same dvd together");
    }
    else if  (qtyOrdered >= MAX_NUMBERS_ORDERED) {
        System.out.print("The cart is full");
    }
    else if (qtyOrdered + 2 >=  MAX_NUMBERS_ORDERED ) {
        System.out.print("the quantity of your adding is too large");
    }
    else {
        itemOrdered[0 + qtyOrdered] =  dvd1;
        itemOrdered[1 + qtyOrdered] = dvd2;
        qtyOrdered += 2;
        System.out.print("Add 2 of dvds sucessfully");
    }

}
```

- 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?

I would prefer array parameter as it is easier to add a certain amount of DVD

Passing parameter

True swap method:

```java
public class TestingPassingParameter {

    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(" hàm swap:");
            System.out.println("jungle dvd title: " + jungleDVD.getTitle());
            System.out.println("cinderella dvd title: " + cinderellaDVD.getTitle()

            changeTitle(jungleDVD, cinderellaDVD.getTitle());

            System.out.println(" hàm changeTitle:");
            System.out.println("jungle dvd title: " + jungleDVD.getTitle());


        }

        public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
            DigitalVideoDisc temp = dvd1;
            dvd1 = dvd2;
            dvd2 = temp;
        }

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


        }

    }
}
```

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

JAVA is a pass by value programming language. The dvd1 and dvd2 in the method are copies of the reference to the original dvd1 and dvd2

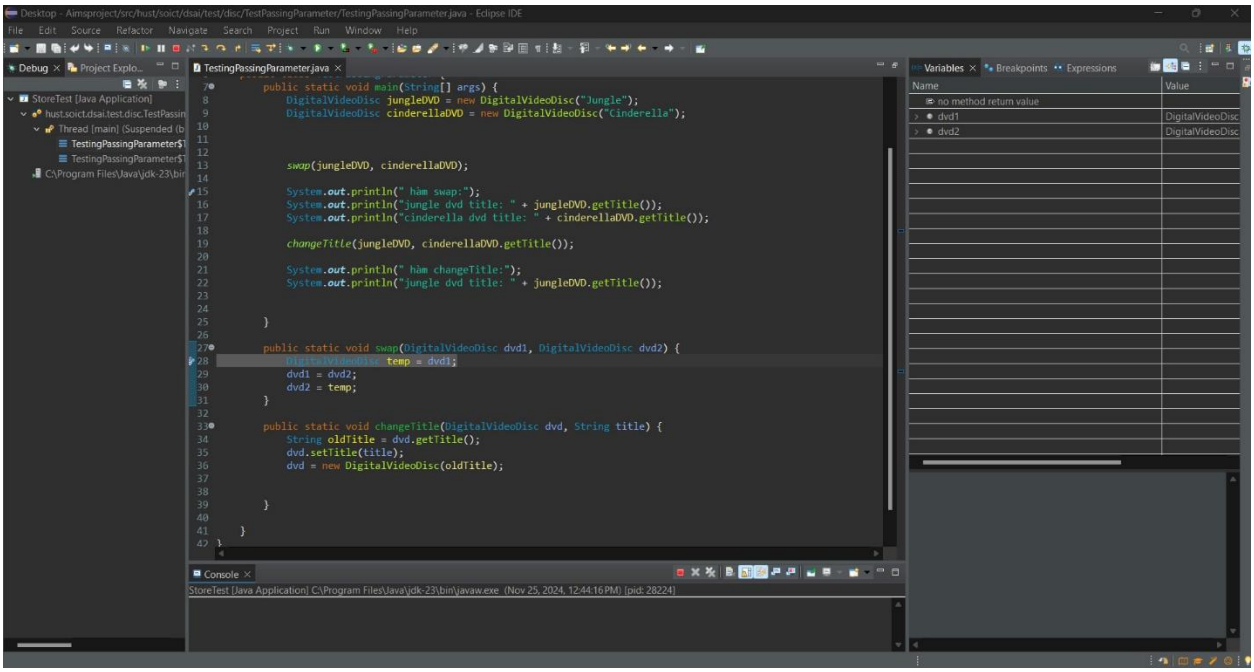
- After the call of **swap(jungleDVD, cinderellaDVD)** why does the title of these two objects still remain?

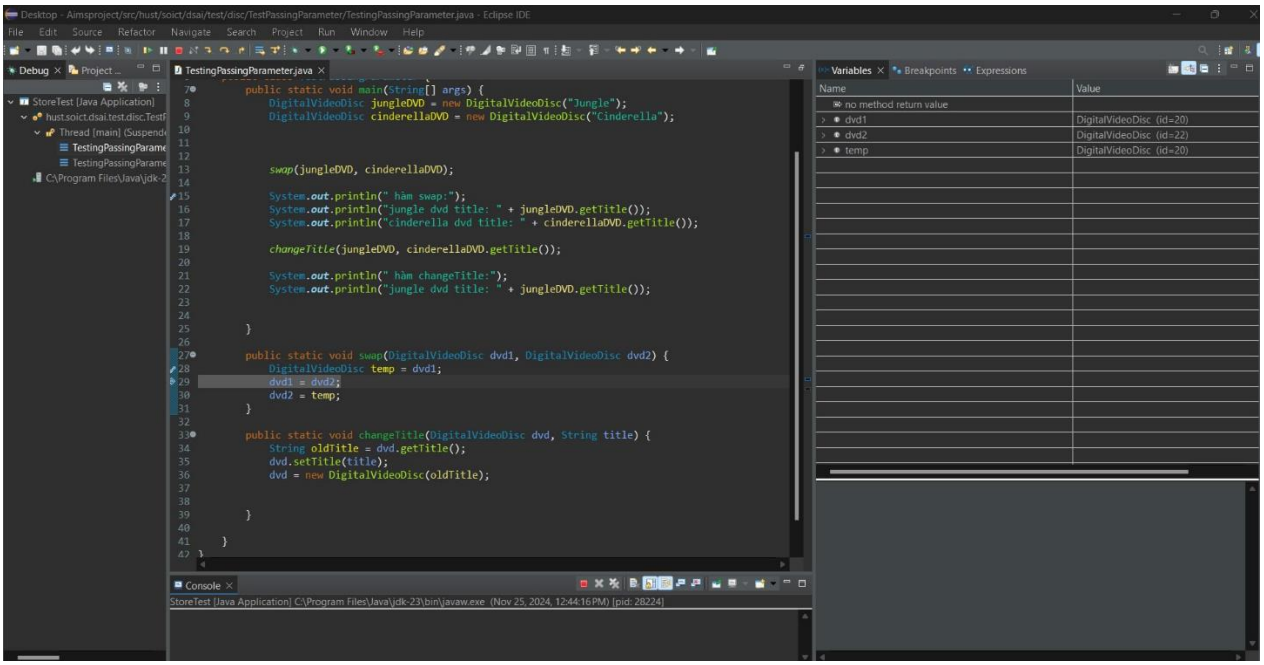
The dvd1 and dvd2 in the method are copies of the reference to the original dvd1 and dvd2, therefore swap only swap the copies, not change the original object

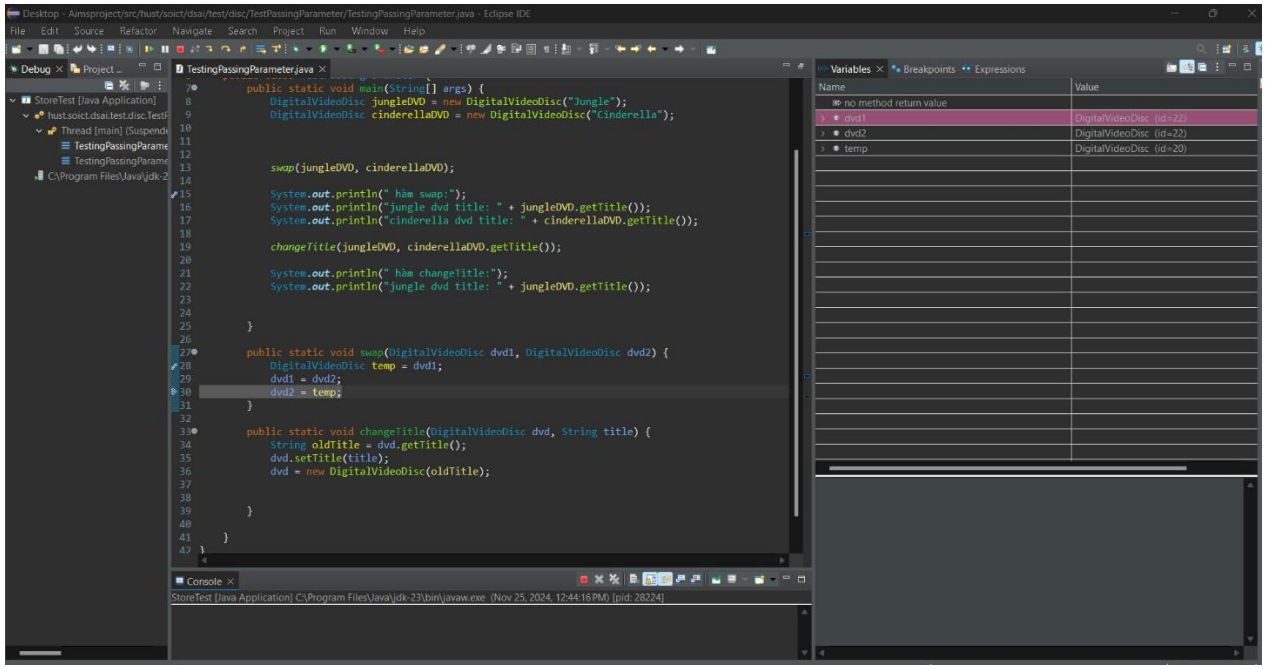
- After the call of **changeTitle(jungleDVD, cinderellaDVD.getTitle())** why is the title of the JungleDVD changed?

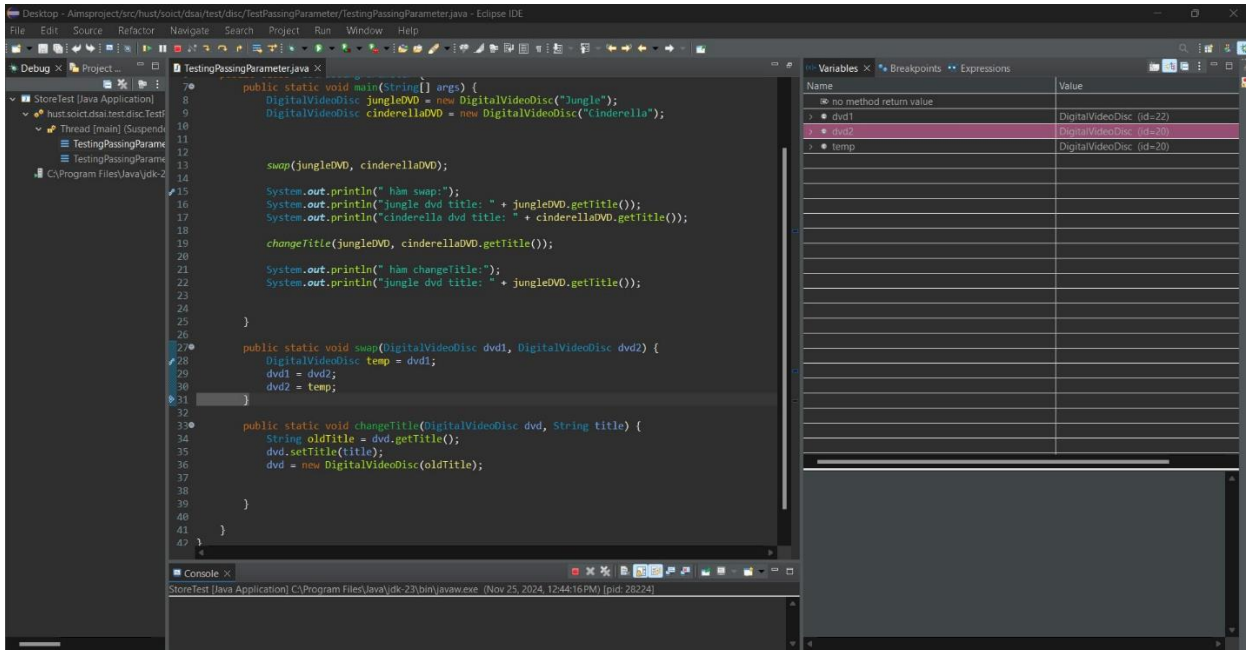
changeTitle take the address and modify title at that address, therefore it also change the original dvd

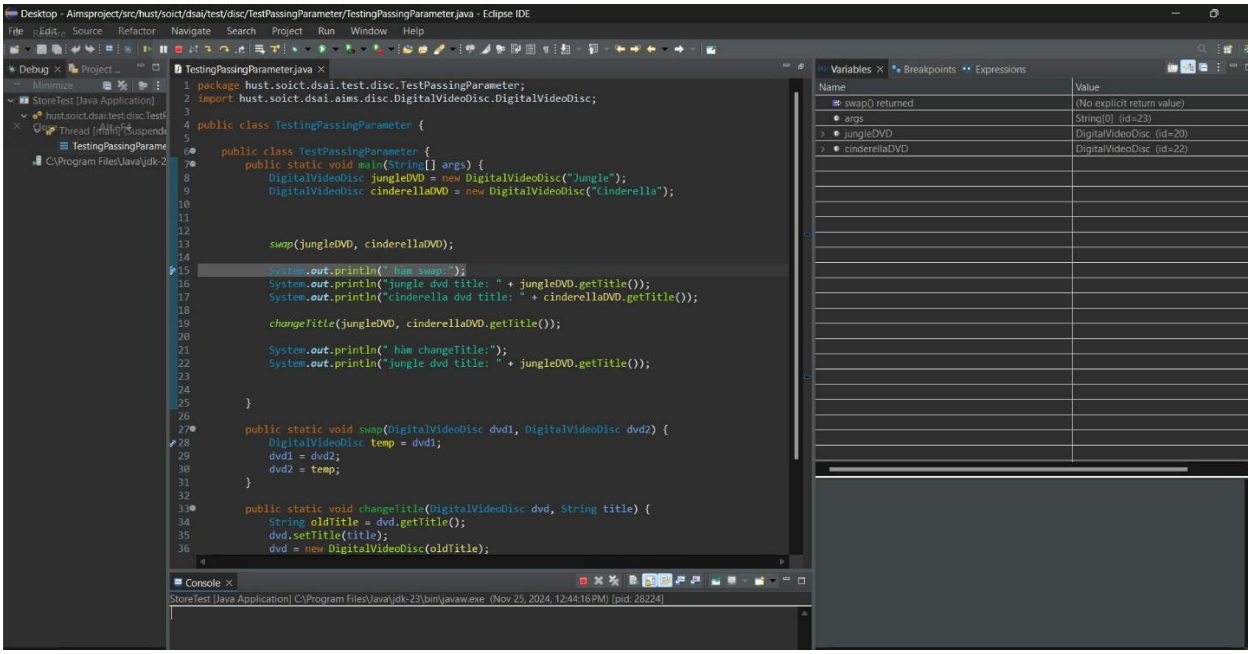
Debug screenshot

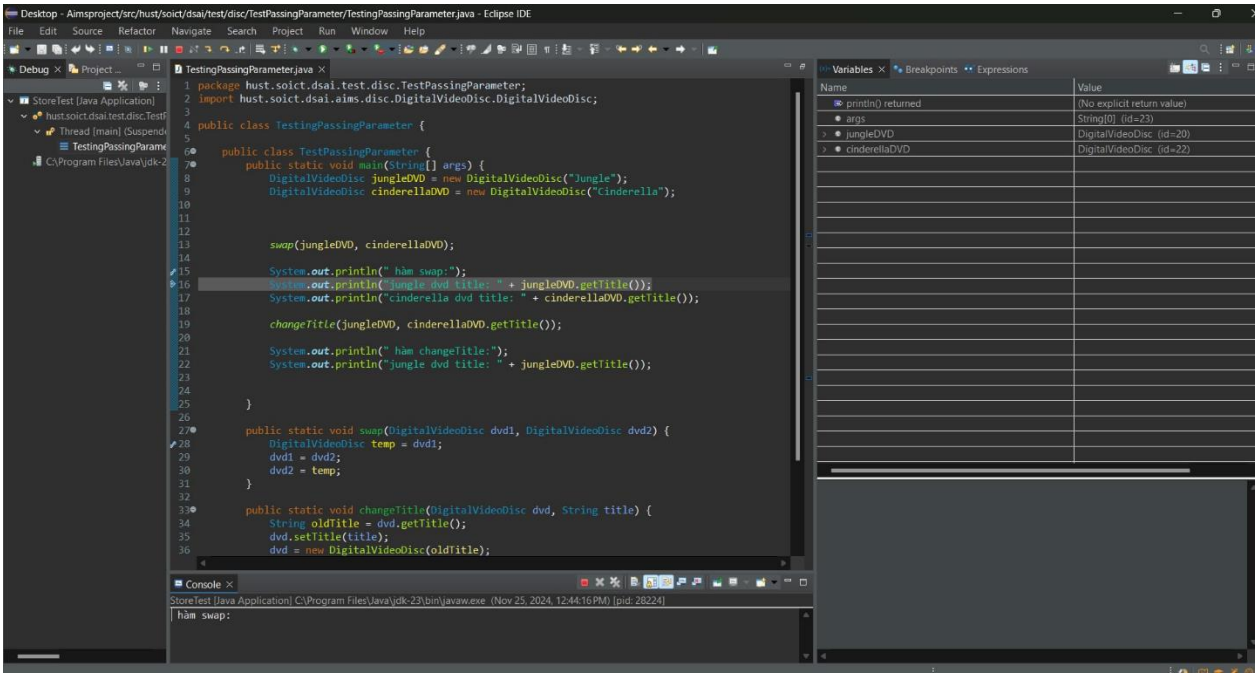


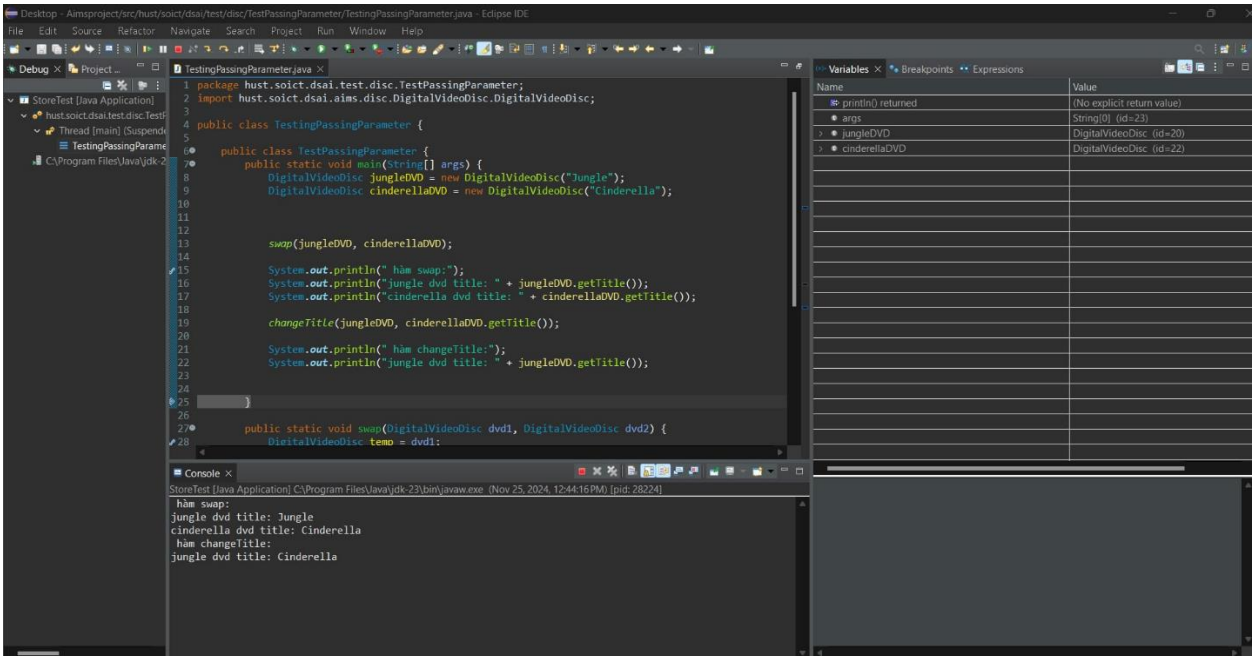



Desktop - Aimsproject/src/hust/soict/dsai/test/disc/TestPassingParameter/TestingPassingParameter.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestingPassingParameter.java
Variables Breakpoints Expressions
Name Value
no method return value
dvd1 DigitalVideoDisc (id=22)
dvd2 DigitalVideoDisc (id=22)
temp DigitalVideoDisc (id=20)
Console
StoreTest [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (Nov 25, 2024, 12:44:16 PM) [pid: 28224]


Desktop - Aimsproject/src/hust/soict/dsai/test/disc/TestPassingParameter/TestingPassingParameter.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestingPassingParameter.java
Variables Breakpoints Expressions
Name Value
no method return value
dvd1 DigitalVideoDisc (id=22)
dvd2 DigitalVideoDisc (id=20)
temp DigitalVideoDisc (id=20)
Console
StoreTest [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (Nov 25, 2024, 12:44:16 PM) [pid: 28224]


Desktop - Aimsproject/src/hust/soict/dsai/test/disc/TestPassingParameter/TestingPassingParameter.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestingPassingParameter.java
package hust.soict.dsai.test.disc.TestPassingParameter;
import hust.soict.dsai.aims.disc.DigitalVideoDisc.DigitalVideoDisc;
public class TestingPassingParameter {
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(" hàm swap:");
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
System.out.println("cinderella dvd title: " + cinderellaDVD.getTitle());
changeTitle(jungleDVD, cinderellaDVD.getTitle());
System.out.println(" hàm changeTitle:");
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
}
public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
DigitalVideoDisc temp = dvd1;
dvd1 = dvd2;
dvd2 = temp;
}
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=23)
jungleDVD
DigitalVideoDisc (id=20)
cinderellaDVD
DigitalVideoDisc (id=22)
Console
StoreTest [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (Nov 25, 2024, 12:44:16 PM) [pid: 28224]


Desktop - Aimsproject/src/hust/soict/dsai/test/disc/TestPassingParameter/TestingPassingParameter.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestingPassingParameter.java
package hust.soict.dsai.test.disc.TestPassingParameter;
import hust.soict.dsai.aims.disc.DigitalVideoDisc.DigitalVideoDisc;
public class TestingPassingParameter {
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(" hàm swap:");
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
System.out.println("cinderella dvd title: " + cinderellaDVD.getTitle());
changeTitle(jungleDVD, cinderellaDVD.getTitle());
System.out.println(" hàm changeTitle:");
System.out.println("jungle dvd title: " + jungleDVD.getTitle());
}
public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
DigitalVideoDisc temp = dvd1;
dvd1 = dvd2;
dvd2 = temp;
}
public static void changeTitle(DigitalVideoDisc dvd, String title) {
String oldTitle = dvd.getTitle();
dvd.setTitle(title);
dvd = new DigitalVideoDisc(oldTitle);
Variables Breakpoints Expressions
Name
Value
println() returned
(No explicit return value)
args
String[0] (id=23)
jungleDVD
DigitalVideoDisc (id=20)
cinderellaDVD
DigitalVideoDisc (id=22)
Console
StoreTest [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (Nov 25, 2024, 12:44:16 PM) [pid: 28224]
hàm swap:



Classifier Member and Instance Member

```java
public class DigitalVideoDisc {

    private String title;
    private String category;
    private String director;
    private int length;
    private float cost;
    private static int nbDigitalVideoDiscs = 0;
    private int id;
```

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

Open the Cart class

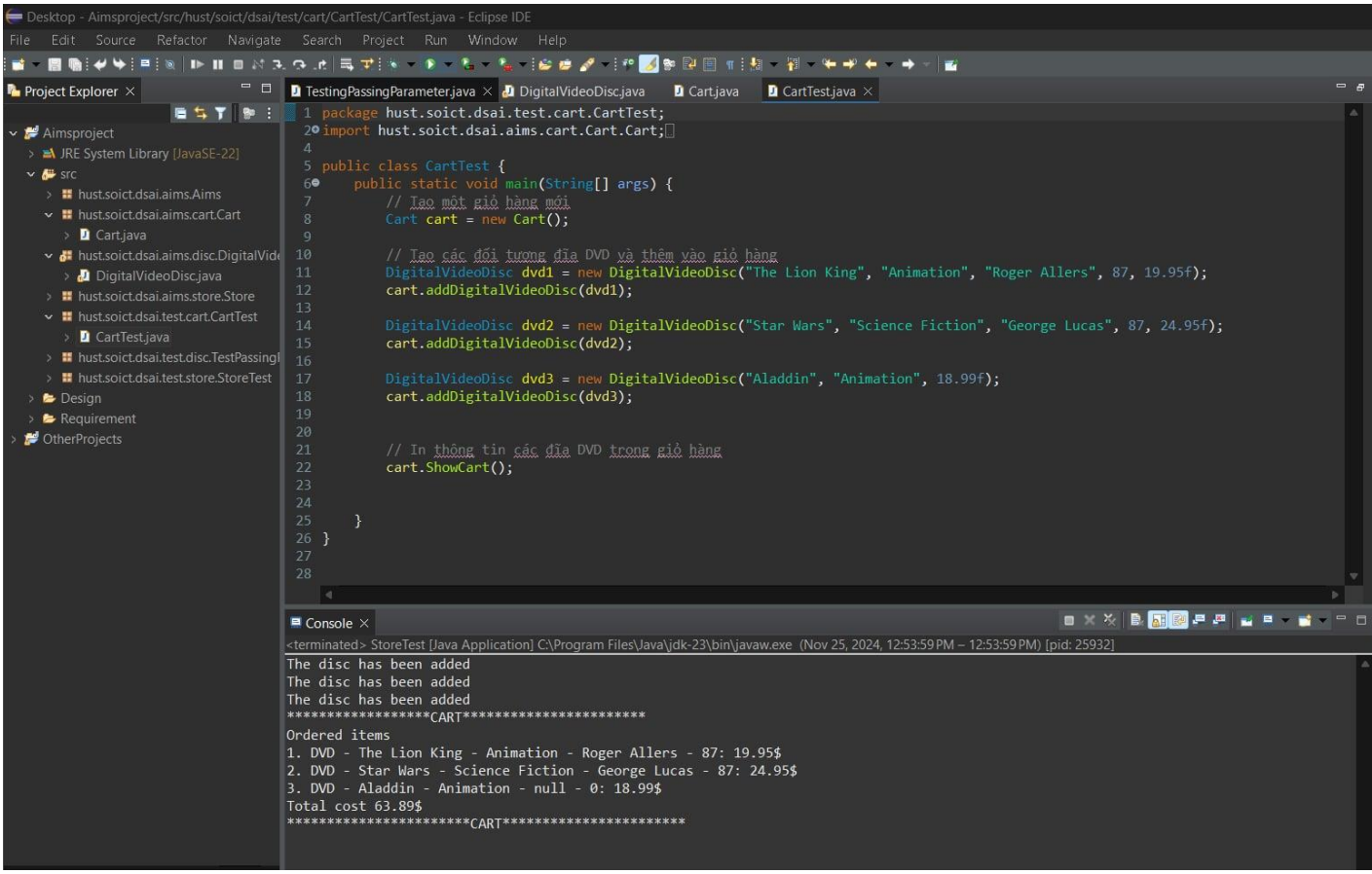
- Write a toString() method for the DigitalVideoDisc class. What should be the return type of this method?

It should be String

```java
public void ShowCart () {
    System.out.println("*****************CART**********************");
    System.out.println("Ordered items");
    for (int i = 0; i < qtyOrdered; i ++ ) {
        System.out.println((i + 1) +". DVD - " + itemOrdered[i].getTitle()+ " - "
    + itemOrdered[i].getCategory() + " - " + itemOrdered[i].getDirector() + " - " +
                itemOrdered[i].getLength() + ": " + itemOrdered[i].getCost() + "$");
    }

    System.out.println("Total cost " + totalCost());
    System.out.println("**********************CART**********************");

}
```



Implement Store class

```java
public class Store {

    public static final int StoreCapacity = 10000;
    private DigitalVideoDisc itemInStore[] = new DigitalVideoDisc[StoreCapacity];

    private int qtyOrdered = 0;

    public void  addDVD(DigitalVideoDisc disc) {
        if  (qtyOrdered >=  StoreCapacity) {
            System.out.println("The Store is full");
        }
        else {
            itemInStore[qtyOrdered] = disc;
            qtyOrdered += 1;
            System.out.println("The disc has been added");
            if (qtyOrdered >=  StoreCapacity - 1) {
                System.out.println("The Store is almost full");
            }

        }

    }
    public void removeDVD (DigitalVideoDisc disc) {
        int index = 0;
        for (int i = 0; i < qtyOrdered; i++) {
            if (itemInStore[i].equals(disc)) {
                index = i;
                break;
            }
        }
        for ( int j = index; j <  qtyOrdered - 1; j++ ) {
            itemInStore[j] = itemInStore[j + 1];

        }
        itemInStore[qtyOrdered - 1] = null;
        qtyOrdered = qtyOrdered - 1;
        System.out.println("Removed sucessfully");

```

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

public class StoreTest {

    public static void main(String[] args) {

        Store store = new Store();

        DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King", "Animation", "Roger Allers", 87, 19.95f);


        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars", "Science Fiction", "George Lucas", 87, 24.95f);


        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladdin", "Animation", 18.99f);


        store.addDVD(dvd1);
        store.addDVD(dvd2);
        store.addDVD(dvd3);

        store.removeDVD(dvd1);
        store.removeDVD(dvd2);
        store.removeDVD(dvd3);

       }
}
```

String, StringBuilder and StringBuffer

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

import java.util.Random;

public class ConcatenationInLoops {
    public static void main(String[] args) {
        Random r = new Random(123);
        long start = System.currentTimeMillis();
        String s = "";
        for (int i = 0; i < 65536; i++)
            s += r.nextInt(2);
        System.out.println(System.currentTimeMillis() - start);

        r = new Random(123);
        start = System.currentTimeMillis();
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < 65536; i++)
            sb.append(r.nextInt(2));
        s = sb.toString();
        System.out.println(System.currentTimeMillis() - start);
    }
}
```

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

import java.nio.file.Files;

public class GarbageCreator {
    public static void main(String[] args) {
        String filename = "text_ofoop.txt" ;
        byte[] inputBytes = { 0 };
        long startTime, endTime;

        try {
            inputBytes = Files.readAllBytes(Paths.get(filename));
            startTime = System.currentTimeMillis();
            String outputString = "";
            for (byte b : inputBytes) {
                outputString += (char) b;
            }
            endTime = System.currentTimeMillis();
            System.out.println(endTime - startTime);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

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

import java.nio.file.Files;

public class NoGarbage {
    public static void main(String[] args) {
        String filename = "text_ofoop.txt";
        byte[] inputBytes = { 0 };
        long startTime, endTime;

        try {
            inputBytes = Files.readAllBytes(Paths.get(filename));
            startTime = System.currentTimeMillis();


            StringBuilder outputStringBuilder = new StringBuilder();
            for (byte b : inputBytes) {
                outputStringBuilder.append((char) b);
            }
            String outputString = outputStringBuilder.toString();

            endTime = System.currentTimeMillis();
            System.out.println("Time taken: " + (endTime - startTime) + " ms");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```