

REPORT LAB 03: BASIC OBJECT-ORIENTED TECHNIQUES

2. Working with some method overloading

New code:

```
20 // New method: Add multiple DVDs using varargs
21 public void addDigitalVideoDisc(DigitalVideoDisc... dvds) {
22     for (DigitalVideoDisc dvd : dvds) {
23         this.addDigitalVideoDisc(dvd);
24     }
25 }
26
27 // New method: Search by title
28 public DigitalVideoDisc searchByTitle(String title) {
29     for (int i = 0; i < qtyOrdered; i++) {
30         if (itemsOrdered[i] != null && itemsOrdered[i].getTitle().equalsIgnoreCase(title)) {
31             return itemsOrdered[i];
32         }
33     }
34     System.out.println("No DVD found with title: " + title);
35     return null;
36 }
37
38 public DigitalVideoDisc searchById(int id) {
39     for (int i = 0; i < qtyOrdered; i++) {
40         if (itemsOrdered[i] != null && itemsOrdered[i].getId() == id) {
41             return itemsOrdered[i];
42         }
43     }
44     System.out.println("No DVD found with ID: " + id);
45     return null;
46 }
47
48 // Update displayCart to improve formatting
49 public void displayCart() {
50     System.out.println("*****CART*****");
51     System.out.println("Ordered Items:");
52     for (int i = 0; i < qtyOrdered; i++) {
53         DigitalVideoDisc dvd = itemsOrdered[i];
54         System.out.println((i + 1) + ". DVD - " + dvd.getTitle() + " - " + dvd.getCategory() +
55             " - " + dvd.getDirector() + " - " + dvd.getLength() + " mins: $" + dvd.getCost());
56     }
57     System.out.println("Total cost: $" + totalCost());
58     System.out.println("*****");
59 }
60
```

Result:

```
The disc has been added.
The disc has been added.
The disc has been added.
*****CART*****
Ordered Items:
1. DVD - The Lion King - Animation - Roger Allers - 87 mins: $19.95
2. DVD - Star Wars - Science Fiction - George Lucas - 124 mins: $24.95
3. DVD - Aladdin - Animation - John Musker - 90 mins: $18.99
Total cost: $63.89
*****
```

3. Passing parameter

```
1 package hust.soict.dsai.test.disc;
2
3 import hust.soict.dsai.aims.disc.DigitalVideoDisc;
4
5 public class TestPassingParameter {
6     Run | Debug | Run main | Debug main
7     public static void main(String[] args) {
8         DigitalVideoDisc dvd1 = new DigitalVideoDisc(title:"Jungle");
9         DigitalVideoDisc dvd2 = new DigitalVideoDisc(title:"Cinderella");
10
11         swap(dvd1, dvd2);
12         System.out.println("After swap:");
13         System.out.println("DVD 1 title: " + dvd1.getTitle());
14         System.out.println("DVD 2 title: " + dvd2.getTitle());
15
16         changeTitle(dvd1, dvd2.getTitle());
17         System.out.println("After changeTitle:");
18         System.out.println("DVD 1 title: " + dvd1.getTitle());
19     }
20
21     public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
22         DigitalVideoDisc temp = dvd1;
23         dvd1 = dvd2;
24         dvd2 = temp;
25
26         System.out.println("Inside swap method:");
27         System.out.println("dvd1: " + dvd1.getTitle());
28         System.out.println("dvd2: " + dvd2.getTitle());
29     }
30
31     public static void changeTitle(DigitalVideoDisc dvd, String title) {
32         dvd.setTitle(title);
33     }
34 }
```

Result:

```
PS D:\java\java_lab2> & 'C:\Program Files (x86)\Java\jre1.8.0_361\bin\java.exe' '-cp' 'D:\java\java_lab2\AimsProject\bin' 'hust.s
oict.dsai.test.disc.TestPassingParameter'
Inside swap method:
dvd1: Cinderella
dvd2: Jungle
After swap:
DVD 1 title: Jungle
DVD 2 title: Cinderella
After changeTitle:
DVD 1 title: Cinderella
```

4. Debug Run: Successful

5. Classifier Member and Instance Member

New code

```
63     public void setTitle(String title) {
64         this.title = title;
65     }

    private double cost;
    private int id;
    private static int nbDigitalVideoDiscs = 0;

    public DigitalVideoDisc(String title) {
        this.title = title;
        this.id = ++nbDigitalVideoDiscs; // Assign a unique ID automatically
    }

    public int getId() {
        return id;
    }

    public static int getNbDigitalVideoDiscs() {
        return nbDigitalVideoDiscs;
    }
```

```
//detailed cart display
public void displayCartDetailed() {
    System.out.println("*****CART DETAILED*****");
    System.out.println("Ordered Items:");
    for (int i = 0; i < qtyOrdered; i++) {
        DigitalVideoDisc dvd = itemsOrdered[i];
        System.out.println("ID: " + dvd.getId());
        System.out.println("Title: " + dvd.getTitle());
        System.out.println("Category: " + dvd.getCategory());
        System.out.println("Director: " + dvd.getDirector());
        System.out.println("Length: " + dvd.getLength() + " minutes");
        System.out.println("Cost: $" + dvd.getCost());
        System.out.println();
    }
    System.out.println("Total cost: $" + totalCost());
    System.out.println("*****");
}
```

```

Displaying cart contents (Detailed):
*****CART DETAILED*****
Ordered Items:
ID: 1
Title: The Lion King
Category: Animation
Director: Roger Allers
Length: 87 minutes
Cost: $19.95

ID: 2
Title: Star Wars
Category: Science Fiction
Director: George Lucas
Length: 124 minutes
Cost: $24.95

ID: 3
Title: Aladdin
Category: Animation
Director: John Musker
Length: 90 minutes
Cost: $18.99

Total cost: $63.89
*****

```

6. Open the Cart class

Update searchbyID and create new CartTest

```

oict.dsai.test.cart.CartTest'
The disc has been added.
The disc has been added.
The disc has been added.
*****CART*****
Ordered Items:
1. DVD - Movie 1 - null - null - 0 mins: $0.0
2. DVD - Movie 2 - null - null - 0 mins: $0.0
3. DVD - Movie 3 - null - null - 0 mins: $0.0
Total cost: $0.0
*****
The disc has been removed.
*****CART*****
Ordered Items:
1. DVD - Movie 2 - null - null - 0 mins: $0.0
2. DVD - Movie 3 - null - null - 0 mins: $0.0
Total cost: $0.0
*****
Searching for DVD with ID 1:
No DVD found with ID: 1
Searching for DVD with title 'Movie 2':
Title: Movie 2
Category: null
Director: null
Length: 0 minutes
Cost: $0.0

```

```

Total cost: $63.89
*****

Searching for DVD with ID: 1
DVD found by ID:
Title: The Lion King
Category: Animation
Director: Roger Allers
Length: 87 minutes
Cost: $19.95

Searching for DVD with title: "The Lion King"
DVD found by title:
Title: The Lion King
Category: Animation
Director: Roger Allers
Length: 87 minutes
Cost: $19.95

Playing a DVD demo:
Playing DVD: The Lion King
DVD length: 87 minutes

The disc has been removed.
*****CART*****
Ordered Items:
1. DVD - The Lion King - Animation - Roger Allers - 87 mins: $19.95
2. DVD - Aladdin - Animation - John Musker - 90 mins: $19.99

```

7. Implement the Store Class

Store.java

```

src > hust > soict > dsai > aims > store > Store.java > Language Support for Java(TM) by Red Hat > Store > ItemsInStore
package hust.soict.dsai.aims.store;

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

public class Store {
    private final ArrayList<DigitalVideoDisc> itemsInStore = new ArrayList<>();

    public void addDVD(DigitalVideoDisc dvd) {
        itemsInStore.add(dvd);
        System.out.println("The disc has been added to the store.");
    }

    public void removeDVD(DigitalVideoDisc dvd) {
        if (itemsInStore.remove(dvd)) {
            System.out.println("The disc has been removed from the store.");
        } else {
            System.out.println("The disc was not found in the store.");
        }
    }

    public void displayStore() {
        System.out.println("Store Inventory:");
        for (DigitalVideoDisc dvd : itemsInStore) {
            dvd.displayInfo();
        }
    }
}

```

StoreTest.java

```
hustProject > src > hust > soict > dsai > test > store > StoreTest.java > Language Support for Java(TM) by Red Hat > StoreTest > main(String[])
1  package hust.soict.dsai.test.store;
2
3  import hust.soict.dsai.aims.store.Store;
4  import hust.soict.dsai.aims.disc.DigitalVideoDisc;
5
6  public class StoreTest {
7      Run | Debug | Run main | Debug main
8      public static void main(String[] args) {
9          // Create a new store
10         Store store = new Store();
11
12         // Hardcoded DVDs to test
13         DigitalVideoDisc dvd1 = new DigitalVideoDisc(title:"The Lion King", category:"Animation", director:"Roger Allers", length:87, cost:19.95);
14         DigitalVideoDisc dvd2 = new DigitalVideoDisc(title:"Star Wars", category:"Science Fiction", director:"George Lucas", length:121, cost:29.95);
15         DigitalVideoDisc dvd3 = new DigitalVideoDisc(title:"Aladdin", category:"Animation", director:"John Musker", length:90, cost:18.99);
16
17         // Test adding DVDs
18         System.out.println("Adding DVDs to the store...");
19         store.addDVD(dvd1); // Add first DVD
20         store.addDVD(dvd2); // Add second DVD
21         store.addDVD(dvd3); // Add third DVD
22
23         // Test displaying DVDs in the store
24         System.out.println("\nDisplaying store inventory:");
25         store.displayStore();
26
27         // Test removing a DVD
28         System.out.println("\nRemoving 'Star Wars' from the store...");
29         store.removeDVD(dvd2);
30
31         // Test displaying DVDs in the store after removal
32         System.out.println("\nDisplaying updated store inventory:");
33         store.displayStore();
34     }
35 }
```

Result

```
The disc has been removed.
*****CART*****
Ordered Items:
1. DVD - The Lion King - Animation - Roger Allers - 87 mins: $19.95
2. DVD - Aladdin - Animation - John Musker - 90 mins: $18.99
Total cost: $38.94
*****
Cannot play DVD: Invalid DVD (invalid length)

o Playing a DVD demo:
Playing DVD: The Lion King
DVD length: 87 minutes

The disc has been removed.
*****CART*****
Ordered Items:
1. DVD - The Lion King - Animation - Roger Allers - 87 mins: $19.95
2. DVD - Aladdin - Animation - John Musker - 90 mins: $18.99
Total cost: $38.94
*****
Cannot play DVD: Invalid DVD (invalid length)
```

```
Category: Animation  
Director: John Musker  
Length: 90 minutes  
Cost: $18.99
```

```
Removing 'Star Wars' from the store...  
The disc has been removed from the store.
```





```
Displaying updated store inventory:  
Store Inventory:  
Title: The Lion King  
Category: Animation  
Director: Roger Allers  
Length: 87 minutes  
Cost: $19.95
```











```
Title: Aladdin  
Category: Animation  
Director: John Musker  
Length: 90 minutes  
Cost: $18.99
```

8. Re-organize your projects

EXPLORER

✓ OPEN EDITORS

✓ JAVA_LAB2    

- ✓ AimsProject ●
 - > Design ●
 - > Requirement ●
 - ✓ src \ hust \ soict \ dsai ●
 - ✓ aims ●
 - ✓ cart ●
 -  Cart.java M
 - ✓ disc ●
 -  DigitalVideoDisc.java M
 - ✓ store ●
 -  Store.java M
 - ✓ test ●
 - ✓ cart ●
 -  CartTest.java M
 - ✓ disc ●
 -  TestPassingParameter.java M
 - ✓ store ●
 -  StoreTest.java U
 -  Aims.java U
 - ≡ answer.txt U
 - ✓ OtherProjects ●
 - lab02 ●
 - ✓ src \ hust \ soict \ dsai ●
 - ✓ garbage ●
 -  ConcatenationInLoops.java U
 -  GarbageCreator.java U
 -  NoGarbage.java U
 - > lab01 ●

> OUTLINE

> TIMELINE

8. String, StringBuilder and StringBuffer

ConcatenationInLoops.java

```
OtherProjects > src > hust > soict > dsai > garbage > ConcatenationInLoops.java > Language Support for Java(TM) by Red Hat > hust\soict\dsai\garbage
1 package hust.soict.dsai.garbage;
2
3 import java.util.Random;
4
5 public class ConcatenationInLoops {
6     Run | Debug | Run main | Debug main
7     public static void main(String[] args) {
8         Random r = new Random(123);
9
10        // Using + operator
11        long start = System.currentTimeMillis();
12        String s = "";
13        for (int i = 0; i < 65536; i++) {
14            s += r.nextInt(2);
15        }
16        System.out.println("Using + operator: " + (System.currentTimeMillis() - start) + "ms");
17        System.out.println("Resulting string: " + s);
18
19        // Using StringBuilder
20        r = new Random(123);
21        start = System.currentTimeMillis();
22        StringBuilder sb = new StringBuilder();
23        for (int i = 0; i < 65536; i++) {
24            sb.append(r.nextInt(2));
25        }
26        s = sb.toString();
27        System.out.println("Using StringBuilder: " + (System.currentTimeMillis() - start) + "ms");
28        System.out.println("Resulting string: " + s);
29
30        // Using char array
31        r = new Random(123);
32        start = System.currentTimeMillis();
33        char[] charArray = new char[65536];
34        for (int i = 0; i < 65536; i++) {
35            charArray[i] = (char) ('0' + r.nextInt(2));
36        }
37        s = new String(charArray);
38        System.out.println("Using char array: " + (System.currentTimeMillis() - start) + "ms");
39        System.out.println("Resulting string: " + s);
40    }
41}
```

NoGarbage.java

```
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 NoGarbage {
8     Run | Debug | Run main | Debug main
9     public static void main(String[] args) {
10         String filename = "src/test.txt";
11         byte[] inputBytes = {0};
12         long startTime, endTime;
13
14         try {
15             inputBytes = Files.readAllBytes(Paths.get(filename));
16         } catch (IOException e) {
17             System.err.println("Error reading file: " + filename);
18             e.printStackTrace();
19         }
20
21         startTime = System.currentTimeMillis();
22         StringBuilder outStringBuilder = new StringBuilder();
23         for (byte b : inputBytes) {
24             outStringBuilder.append((char) b);
25         }
26         endTime = System.currentTimeMillis();
27         System.out.println("Execution time: " + (endTime - startTime) + "ms");
28         System.out.println(outStringBuilder.toString());
29     }
30 }
```

GarbageCreator.java

```
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     Run | Debug | Run main | Debug main
9     public static void main(String[] args) {
10         String filename = "src/test.txt";
11         byte[] inputBytes = {0};
12         long startTime, endTime;
13
14         try {
15             inputBytes = Files.readAllBytes(Paths.get(filename));
16         } catch (IOException e) {
17             System.err.println("Error reading file: " + filename);
18             e.printStackTrace();
19         }
20
21         startTime = System.currentTimeMillis();
22         String outputString = "";
23         for (byte b : inputBytes) {
24             outputString += (char) b;
25         }
26         endTime = System.currentTimeMillis();
27         System.out.println("Execution time: " + (endTime - startTime) + "ms");
28         System.out.println(outputString);
29     }
30 }
```

Result

```
1001000100111001001110000101011100
Using StringBuilder: 3ms
Using char array: 1ms
PS D:\java\java_lab2>
• PS D:\java\java_lab2> & 'C:\Program Files (x86)\Java\jre1.8.0_361\bin\java.exe' '-cp' 'D:\java\java_lab2\AimsProject\bin' 'hust.s
oict.dsai.garbage.ConcatenationInLoops'
Using + operator: 1242ms
```

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

Java is a passed by value programming language.

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

After calling swap(jungleDVD, cinderellaDVD), the titles remain unchanged because the method only swaps the title field values, not the object references.

The jungleDVD and cinderellaDVD references still point to the same objects, so the objects themselves remain the same, with only their title fields swapped.

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

The title of jungleDVD changes because changeTitle() directly modifies its title field using the value from cinderellaDVD.getTitle().

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

The return type should be String.