

**Đại học Bách khoa Hà Nội**  
**Trường Công nghệ thông tin và Truyền thông**



## **OOP LAB05:**

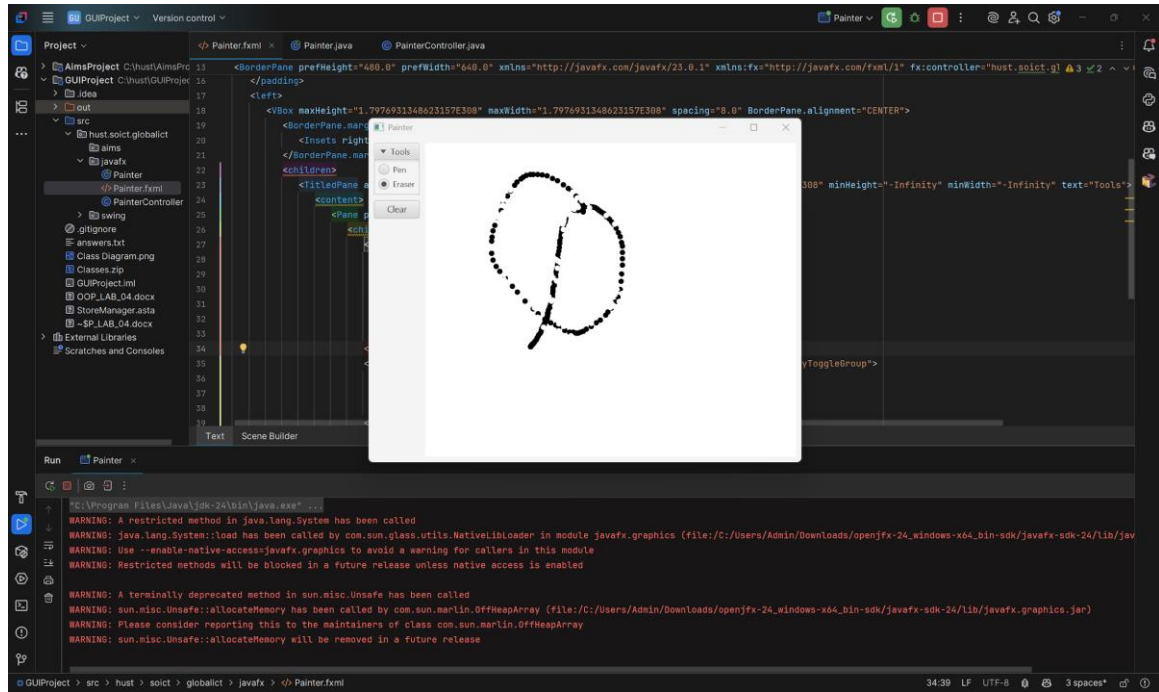
# **GUI Programming with JavaFX and Exception Handling**

**Họ và tên: Lê Thành Nguyên**  
**Giảng viên: Hồ Viết Đức Lương**  
**Mã lớp: 750867**

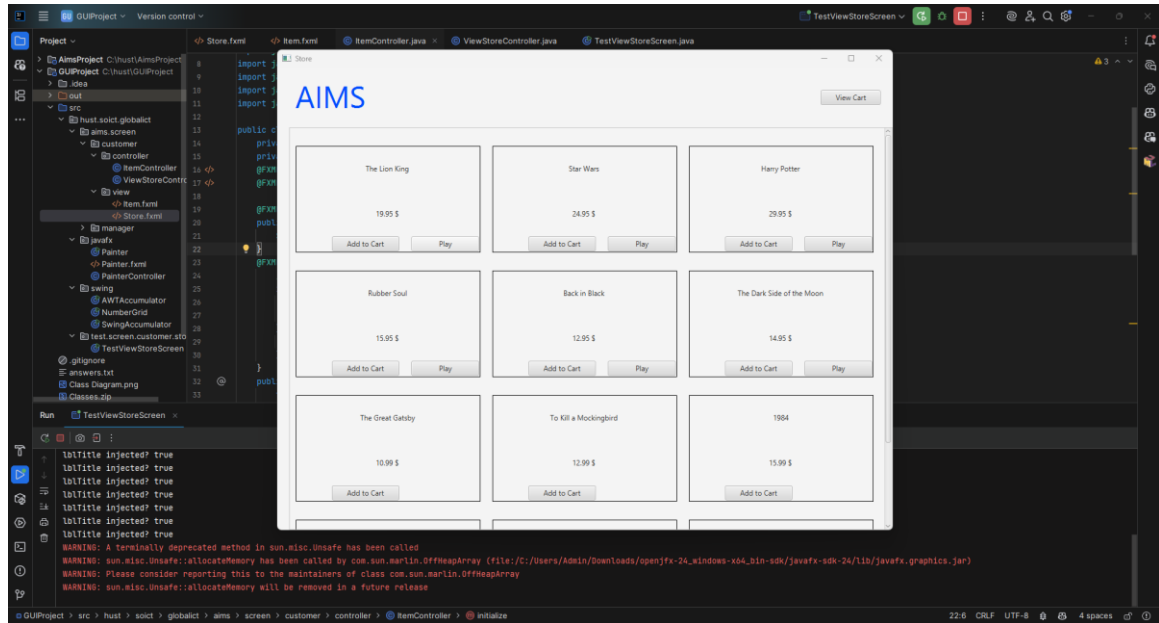
## Nội dung

Exercise 3	3
Exercise 5	3
Exercise 6 & 7	4
Exercise 8	4
Exercise 10	5
Exercise 11	5
Exercise 12	6
Class Diagram	6

### Exercise 3



### Exercise 5



## Exercise 6 & 7

The screenshot shows an IDE with a project named 'GUIProject'. The 'Cart' window is open, displaying a table with the following data:

ID	Title	Category	Cost
1	The Lion King	Animation	19.95
2	Star Wars	Science Fiction	24.95

Below the table, the text 'Total: 0 \$' is displayed. A blue button labeled 'Place Order' is at the bottom. The 'Run' console shows several warnings and messages, including 'Removed: Book 1' and 'Removed: Book 2'.

## Exercise 8

The screenshot shows the same IDE with the 'Cart' window. The table now contains two items:

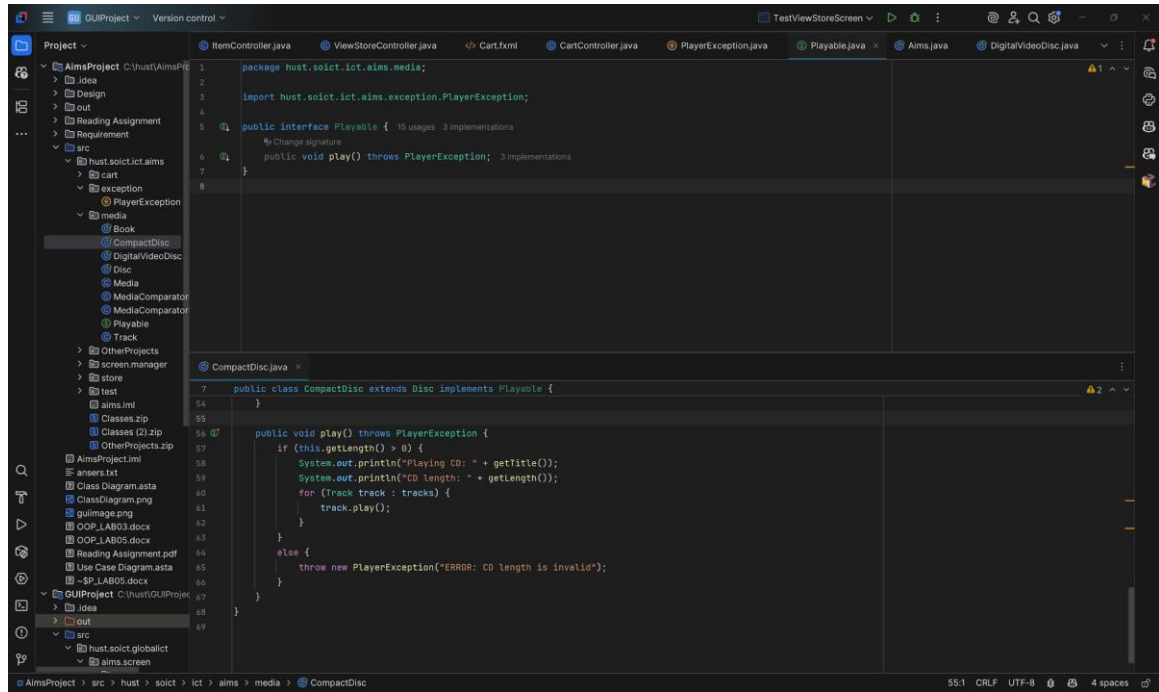
ID	Title	Category	Cost
1	Book 2	Author 2	15.0
2			

The text 'Total: 15.0 \$' is displayed. A blue button labeled 'Place Order' is at the bottom. The 'Run' console shows a detailed list of 'Ordered Items' and their costs:

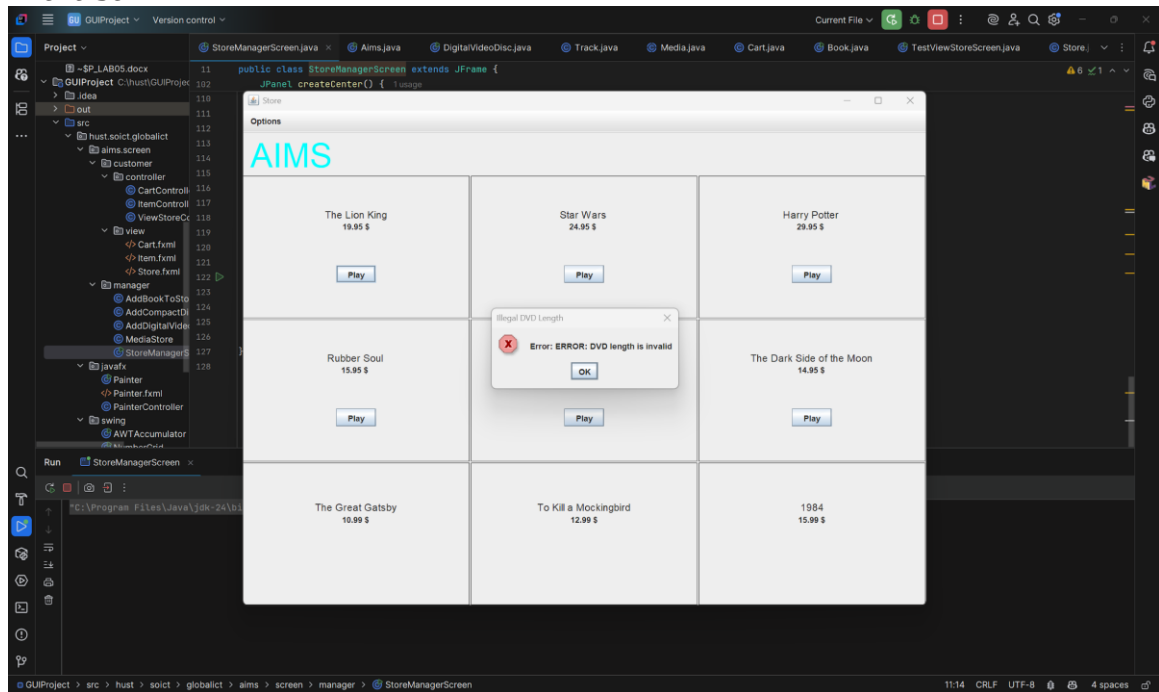
```

Ordered Items:
Book [Title=Book 1, category=Author 1] 10.00
Book [Title=Book 2, category=Author 2] 15.00
Total cost is 25.0$
1 Book 1 10.00
2 Book 2 15.00
  
```

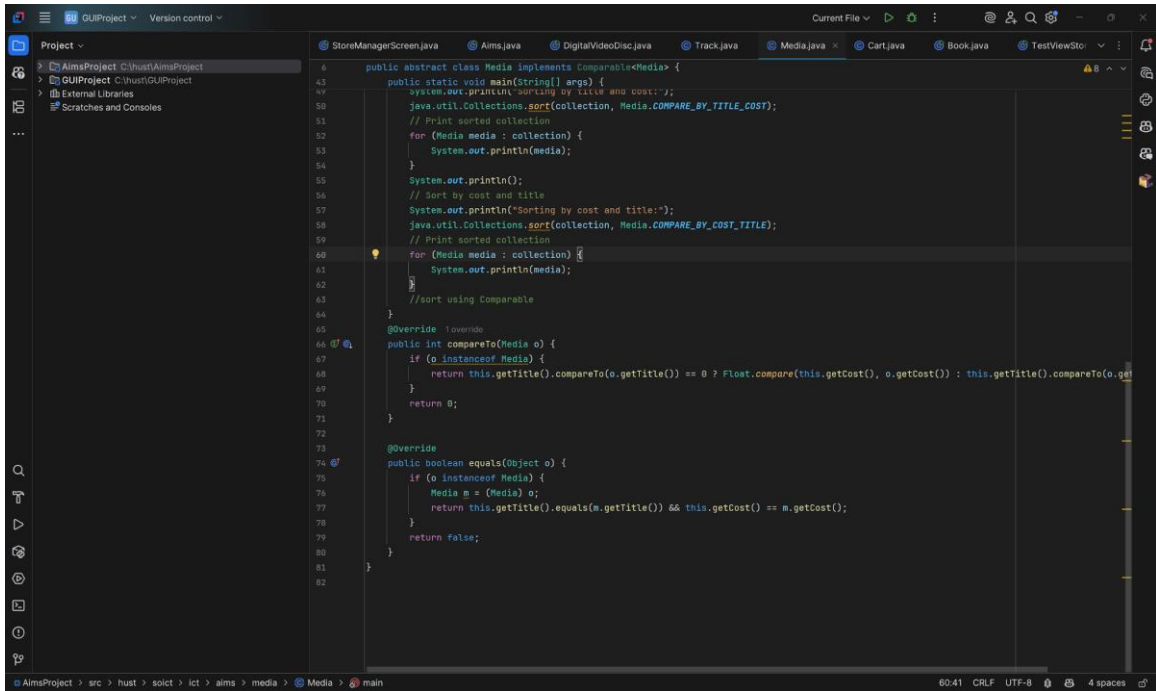
## Exercise 10



## Exercise 11



## Exercise 12



The screenshot shows an IDE with a project named 'AimsProject'. The file explorer on the left shows the project structure. The main editor displays the code for 'Media.java'. The code defines an abstract class 'Media' that implements 'Comparable<Media>'. It includes a 'main' method for testing sorting and two overridden methods: 'compareTo' and 'equals'.

```
6 public abstract class Media implements Comparable<Media> {
43     public static void main(String[] args) {
44         System.out.println("Sorting by title and cost:");
50         java.util.Collections.sort(collection, Media.COMPARE_BY_TITLE_COST);
51         // Print sorted collection
52         for (Media media : collection) {
53             System.out.println(media);
54         }
55         System.out.println();
56         // Sort by cost and title
57         System.out.println("Sorting by cost and title:");
58         java.util.Collections.sort(collection, Media.COMPARE_BY_COST_TITLE);
59         // Print sorted collection
60         for (Media media : collection) {
61             System.out.println(media);
62         }
63         //sort using Comparable
64     }
65
66     @Override
67     public int compareTo(Media o) {
68         if (o instanceof Media) {
69             return this.getTitle().compareTo(o.getTitle()) == 0 ? Float.compare(this.getCost(), o.getCost()) : this.getTitle().compareTo(o.getTitle());
70         }
71         return 0;
72     }
73
74     @Override
75     public boolean equals(Object o) {
76         if (o instanceof Media) {
77             Media m = (Media) o;
78             return this.getTitle().equals(m.getTitle()) && this.getCost() == m.getCost();
79         }
80         return false;
81     }
82 }
```

## Class Diagram

