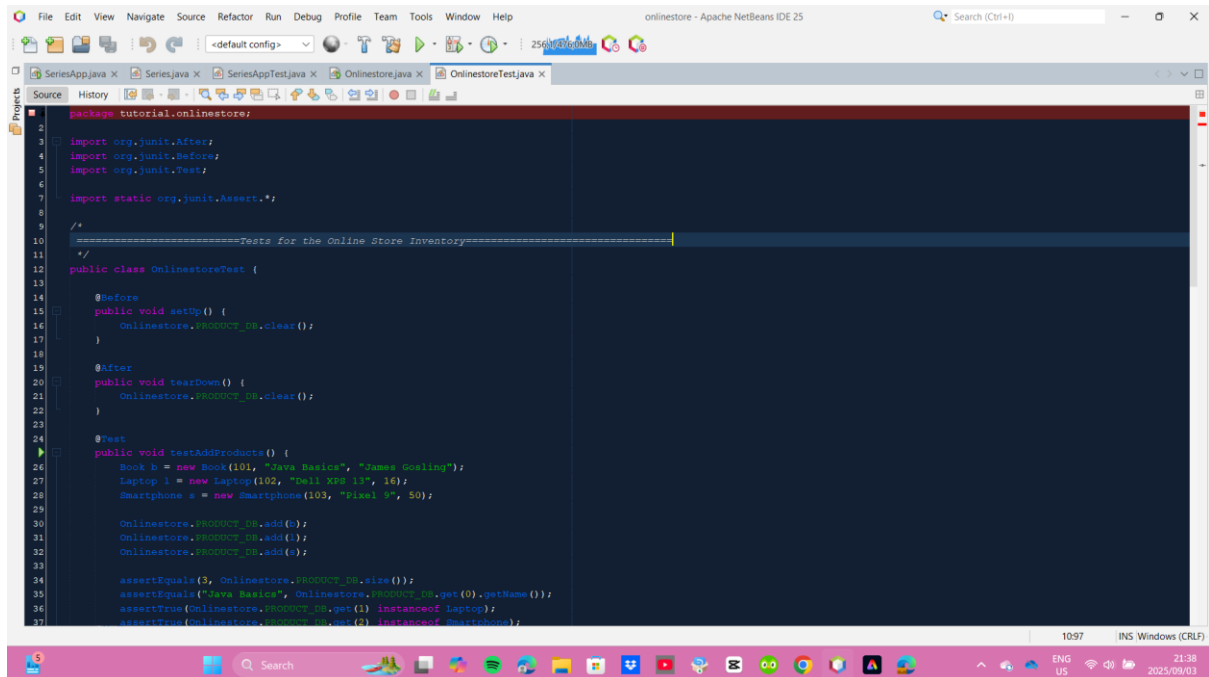


# PROG ASSIGNMENT

## DIVINE .Z.SULUBIKA -ST10485027(GR.3)

### SECTION A :

### MAIN CLASS: SeriesApp.Java



```
package tutorial.onlinestore;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

import static org.junit.Assert.*;

/**
 * =====Tests for the Online Store Inventory=====
 */
public class OnlineStoreTest {

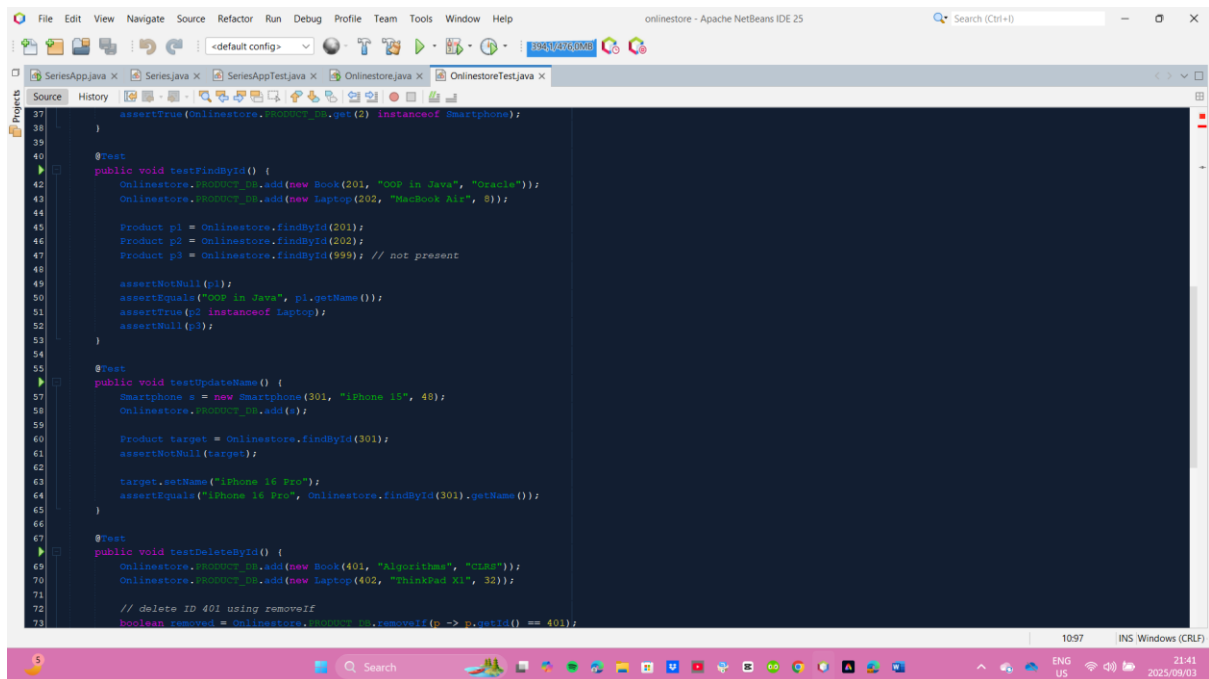
    @Before
    public void setUp() {
        OnlineStore.PRODUCT_DB.clear();
    }

    @After
    public void tearDown() {
        OnlineStore.PRODUCT_DB.clear();
    }

    @Test
    public void testAddProducts() {
        Book b = new Book(101, "Java Basics", "James Gosling");
        Laptop l = new Laptop(102, "Dell XPS 13", 14);
        Smartphone s = new Smartphone(103, "Pixel 3", 50);

        OnlineStore.PRODUCT_DB.add(b);
        OnlineStore.PRODUCT_DB.add(l);
        OnlineStore.PRODUCT_DB.add(s);

        assertEquals(3, OnlineStore.PRODUCT_DB.size());
        assertEquals("Java Basics", OnlineStore.PRODUCT_DB.get(0).getName());
        assertTrue(OnlineStore.PRODUCT_DB.get(1) instanceof Laptop);
        assertTrue(OnlineStore.PRODUCT_DB.get(2) instanceof Smartphone);
    }
}
```



```
        assertTrue(OnlineStore.PRODUCT_DB.get(2) instanceof Smartphone);
    }

    @Test
    public void testFindById() {
        OnlineStore.PRODUCT_DB.add(new Book(201, "OOP in Java", "Delella"));
        OnlineStore.PRODUCT_DB.add(new Laptop(202, "MacBook Air", 8));

        Product p1 = OnlineStore.findById(201);
        Product p2 = OnlineStore.findById(202);
        Product p3 = OnlineStore.findById(999); // not present

        assertNotNull(p1);
        assertEquals("OOP in Java", p1.getName());
        assertTrue(p2 instanceof Laptop);
        assertNull(p3);
    }

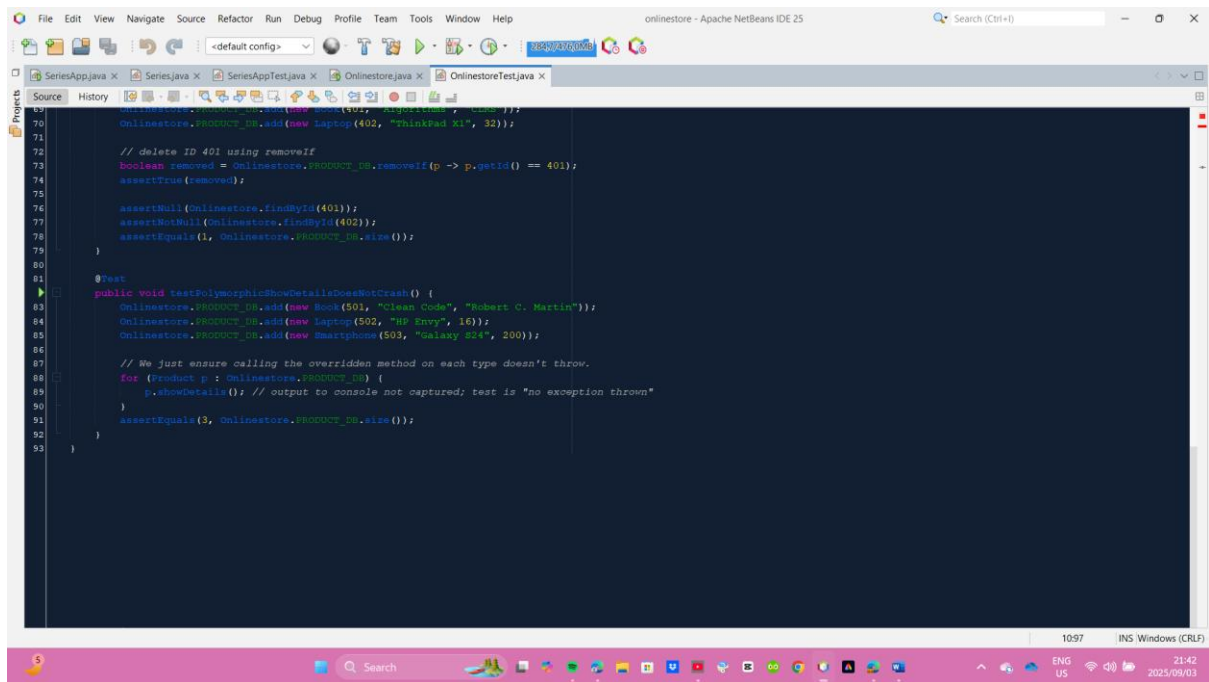
    @Test
    public void testUpdateName() {
        Smartphone s = new Smartphone(301, "iPhone 15", 48);
        OnlineStore.PRODUCT_DB.add(s);

        Product target = OnlineStore.findById(301);
        assertNotNull(target);

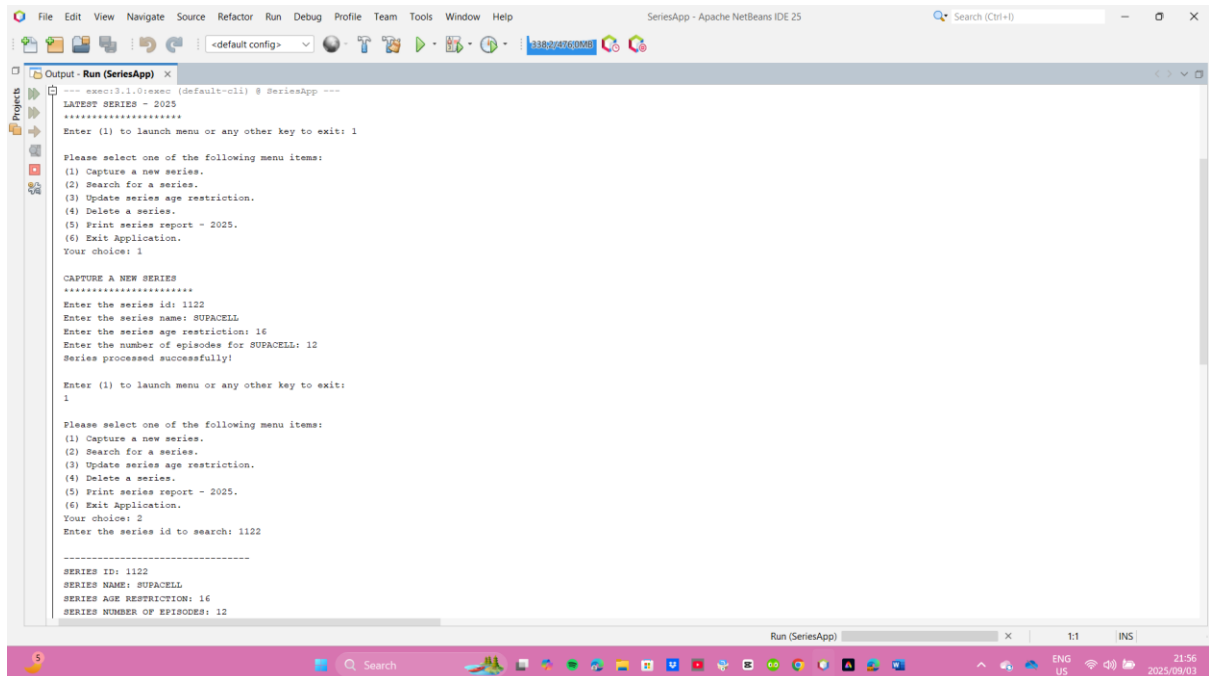
        target.setName("iPhone 16 Pro");
        assertEquals("iPhone 16 Pro", OnlineStore.findById(301).getName());
    }

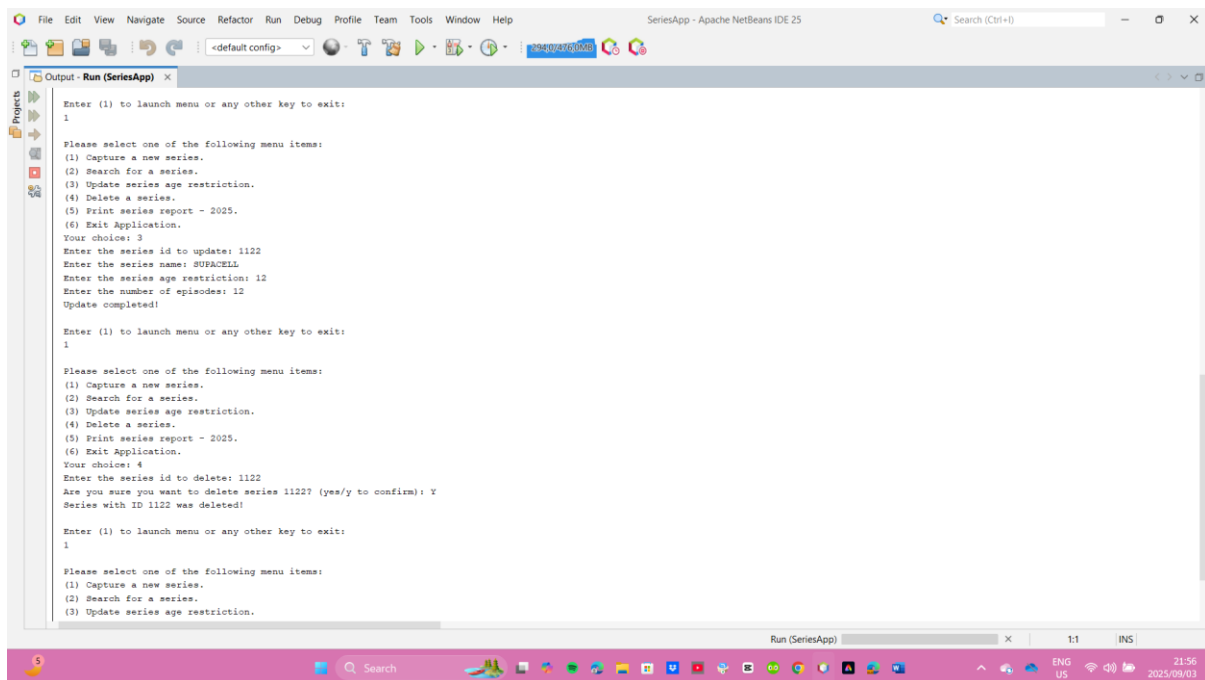
    @Test
    public void testDeleteById() {
        OnlineStore.PRODUCT_DB.add(new Book(401, "Algorithms", "CLRS"));
        OnlineStore.PRODUCT_DB.add(new Laptop(402, "ThinkPad X1", 32));

        // delete ID 401 using removeIf
        boolean removed = OnlineStore.PRODUCT_DB.removeIf(p -> p.getId() == 401);
    }
}
```



## OUTPUT:





Please select one of the following menu items:

- (1) Capture a new series.
- (2) Search for a series.
- (3) Update series age restriction.
- (4) Delete a series.
- (5) Print series report - 2025.
- (6) Exit Application.

Your choice: 5

No series captured yet.

Enter (1) to launch menu or any other key to exit:

||

## CODE:

### MAIN CLASS

//SECTION A :

```
package seriesapp;
```

```
import java.util.ArrayList;
```

```
import java.util.Iterator;
```

```
import java.util.List;

import java.util.Scanner;


public class SeriesApp {

    public static final Scanner input = new Scanner(System.in);

    public static final List<Series> SERIES_DB = new ArrayList<>();


    public static void main(String[] args) {

        System.out.println("LATEST SERIES - 2025");

        System.out.println("*****");

        System.out.print("Enter (1) to launch menu or any other key to exit: ");

        String launch = input.nextLine();


        if (!"1".equals(launch)) {

            System.out.println("Goodbye!");

            return;

        }


        boolean running = true;

        while (running) {

            printMenu();

            String choice = input.nextLine();


            switch (choice) {

                case "1" -> captureNewSeries();

                case "2" -> searchSeriesById();

                case "3" -> updateSeries();

                case "4" -> deleteSeries();

            }

        }

    }

}
```

```

        case "5" -> printReport();
        case "6" -> {
            System.out.println("Exiting application...");
            running = false;
        }
        default -> System.out.println("Invalid option! Please choose 1-6.");
    }

    if (running) {
        System.out.println("\nEnter (1) to launch menu or any other key to exit:");
        String again = input.nextLine();
        if (!"1".equals(again)) {
            System.out.println("Goodbye!");
            break;
        }
    }
}
}

```

// 1.1) Menu

```

public static void printMenu() {
    System.out.println("\nPlease select one of the following menu items:");
    System.out.println("(1) Capture a new series.");
    System.out.println("(2) Search for a series.");
    System.out.println("(3) Update series age restriction.");
    System.out.println("(4) Delete a series.");
    System.out.println("(5) Print series report - 2025.");
    System.out.println("(6) Exit Application.");
}

```

```
System.out.print("Your choice: ");  
}
```

```
// 1.2 & 1.3) Capture new series
```

```
public static void captureNewSeries() {  
    System.out.println("\nCAPTURE A NEW SERIES");  
    System.out.println("*****");  
  
    int id = readInt("Enter the series id: ");  
    System.out.print("Enter the series name: ");  
    String name = input.nextLine();  
    int age = readAgeRestrictionValidated();  
    int episodes = readInt("Enter the number of episodes for " + name + ": ");  
  
    SERIES_DB.add(new Series(id, name, age, episodes));  
    System.out.println("Series processed successfully!");  
}
```

```
public static int readAgeRestrictionValidated() {  
    while (true) {  
        System.out.print("Enter the series age restriction: ");  
        String ageStr = input.nextLine();  
        if (!isInteger(ageStr)) {  
            System.out.println("You have entered an incorrect series age! Please re-enter.");  
            continue;  
        }  
        int age = Integer.parseInt(ageStr);  
        if (age < 2 || age > 18) {
```

```

        System.out.println("You have entered an incorrect series age! Please re-enter.");
        continue;
    }
    return age;
}
}

```

// 1.5) Search by ID

```

public static void searchSeriesById() {
    int id = readInt("Enter the series id to search: ");
    Series s = findById(id);
    if (s == null) {
        System.out.println("\n-----");
        System.out.println("Series with ID " + id + " was not found!");
        System.out.println("-----");
    } else {
        System.out.println("\n-----");
        System.out.println("SERIES ID: " + s.getId());
        System.out.println("SERIES NAME: " + s.getName());
        System.out.println("SERIES AGE RESTRICTION: " + s.getAgeRestriction());
        System.out.println("SERIES NUMBER OF EPISODES: " + s.getEpisodes());
        System.out.println("-----");
    }
}
}

```

// 1.6) Update

```

public static void updateSeries() {
    int id = readInt("Enter the series id to update: ");

```

```
Series s = findById(id);  
if (s == null) {  
    System.out.println("Series with ID " + id + " was not found!");  
    return;  
}
```

```
System.out.print("Enter the series name: ");  
String newName = input.nextLine();  
int newAge = readAgeRestrictionValidated();  
int newEpisodes = readInt("Enter the number of episodes: ");
```

```
s.setName(newName);  
s.setAgeRestriction(newAge);  
s.setEpisodes(newEpisodes);
```

```
System.out.println("Update completed!");  
}
```

// 1.7) Delete

```
public static void deleteSeries() {  
    int id = readInt("Enter the series id to delete: ");  
    Series s = findById(id);  
    if (s == null) {  
        System.out.println("Series with ID " + id + " was not found!");  
        return;  
    }  
}
```



```

        System.out.print("Are you sure you want to delete series " + id + "? (yes/y to confirm):
");
        String confirm = input.nextLine().toLowerCase();
        if (confirm.equals("y") || confirm.equals("yes")) {
            Iterator<Series> it = SERIES_DB.iterator();
            while (it.hasNext()) {
                if (it.next().getId() == id) {
                    it.remove();
                    break;
                }
            }
            System.out.println("Series with ID " + id + " was deleted!");
        } else {
            System.out.println("Delete cancelled.");
        }
    }
}

```

// 1.8) Report

```

public static void printReport() {
    if (SERIES_DB.isEmpty()) {
        System.out.println("No series captured yet.");
        return;
    }

    for (int i = 0; i < SERIES_DB.size(); i++) {
        Series s = SERIES_DB.get(i);
        System.out.println("\nSeries " + (i + 1));
        System.out.println("-----");
    }
}

```

```

        System.out.println("SERIES ID: " + s.getId());
        System.out.println("SERIES NAME: " + s.getName());
        System.out.println("SERIES AGE RESTRICTION: " + s.getAgeRestriction());
        System.out.println("NUMBER OF EPISODES: " + s.getEpisodes());
        System.out.println("-----");
    }
}

```

// Helpers

```

public static Series findById(int id) {
    for (Series s : SERIES_DB) {
        if (s.getId() == id) return s;
    }
    return null;
}

```

```

public static int readInt(String prompt) {
    while (true) {
        System.out.print(prompt);
        String str = input.nextLine();
        if (isInteger(str)) {
            return Integer.parseInt(str);
        }
        System.out.println("Please enter a valid number.");
    }
}

```

```

public static boolean isInteger(String s) {

```

```

    if (s == null || s.isEmpty()) return false;

    try{

        Integer.parseInt(s);

        return true;

    } catch (NumberFormatException e) {

        return false;

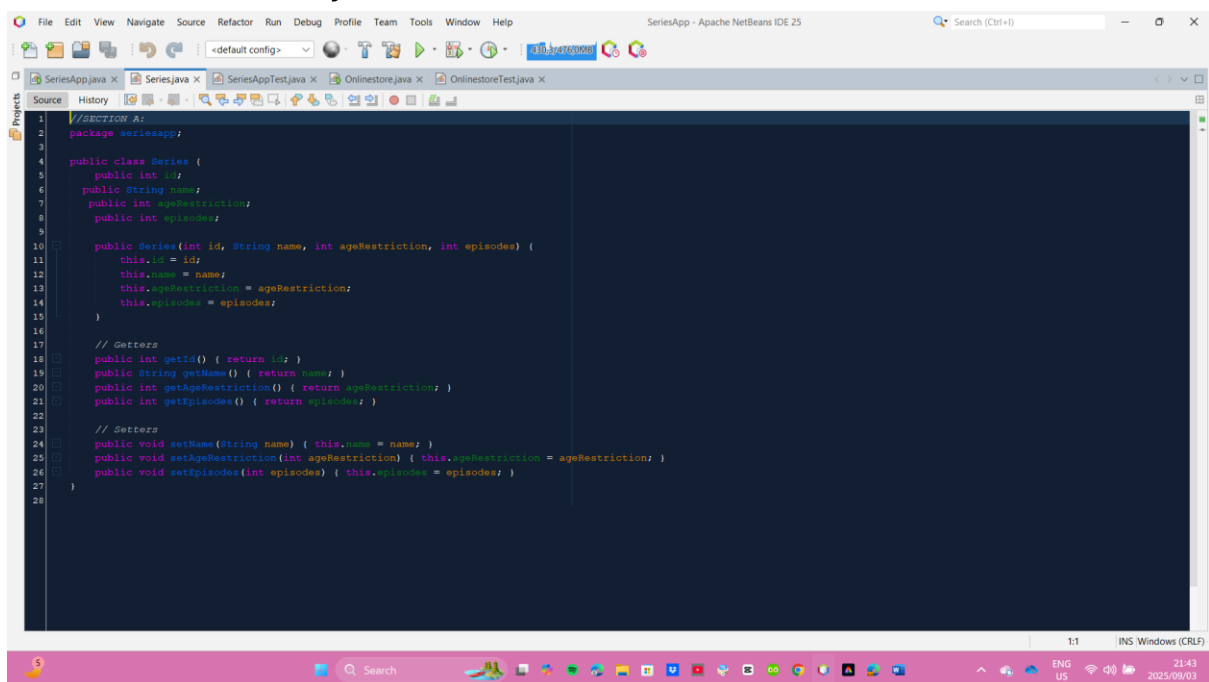
    }

}

}

```

## SERIES CLASS: Series.java



## CODE: SERIES CLASS

```
//SECTION A:
```

```
package seriesapp;
```

```
public class Series {
```

```
    public int id;
```

```
    public String name;
```

```
    public int ageRestriction;
```

```
    public int episodes;
```

```
    public Series(int id, String name, int ageRestriction, int episodes) {
```

```
        this.id = id;
```

```
        this.name = name;
```

```
        this.ageRestriction = ageRestriction;
```

```
        this.episodes = episodes;
```

```
    }
```

```
    // Getters
```

```
    public int getId() { return id; }
```

```
    public String getName() { return name; }
```

```
    public int getAgeRestriction() { return ageRestriction; }
```

```
    public int getEpisodes() { return episodes; }
```

```
    // Setters
```

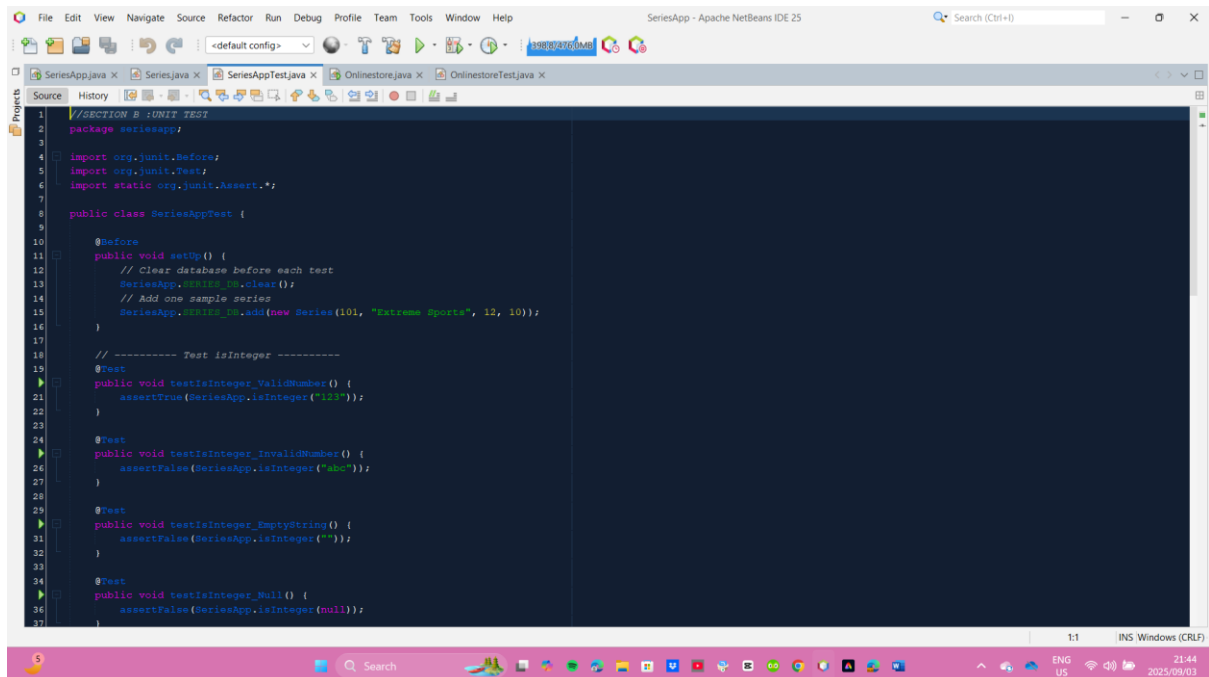
```
    public void setName(String name) { this.name = name; }
```

```
    public void setAgeRestriction(int ageRestriction) { this.ageRestriction =  
ageRestriction; }
```

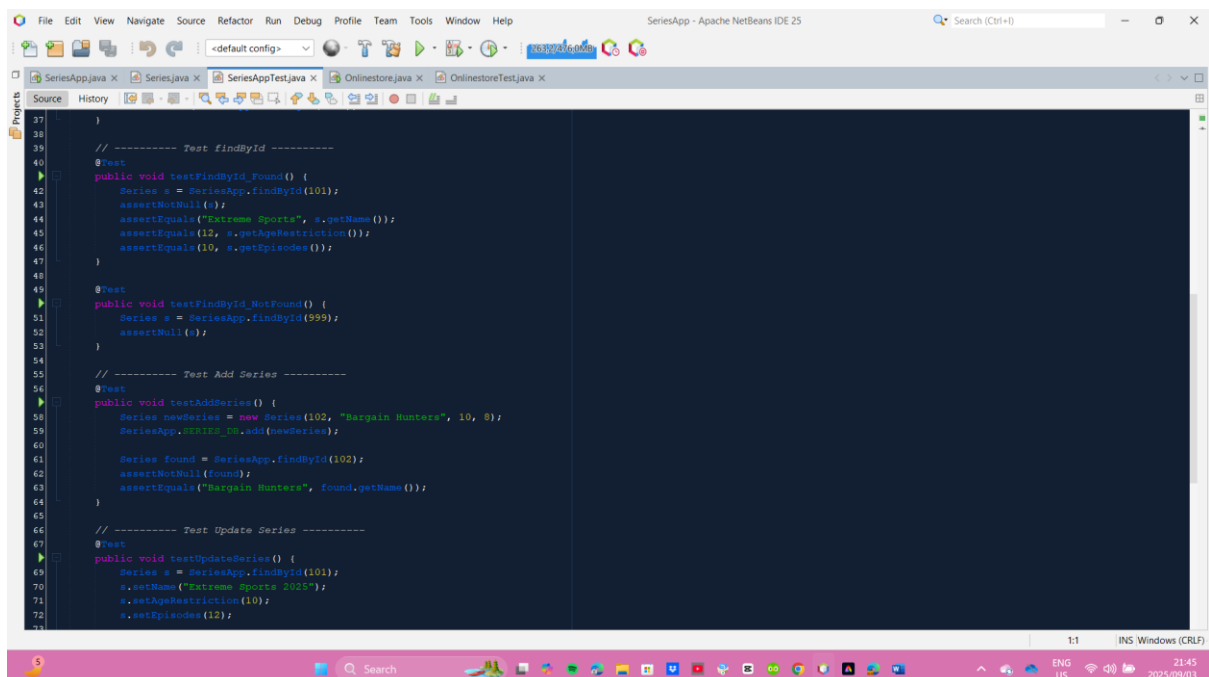
```
    public void setEpisodes(int episodes) { this.episodes = episodes; }
```

```
}
```

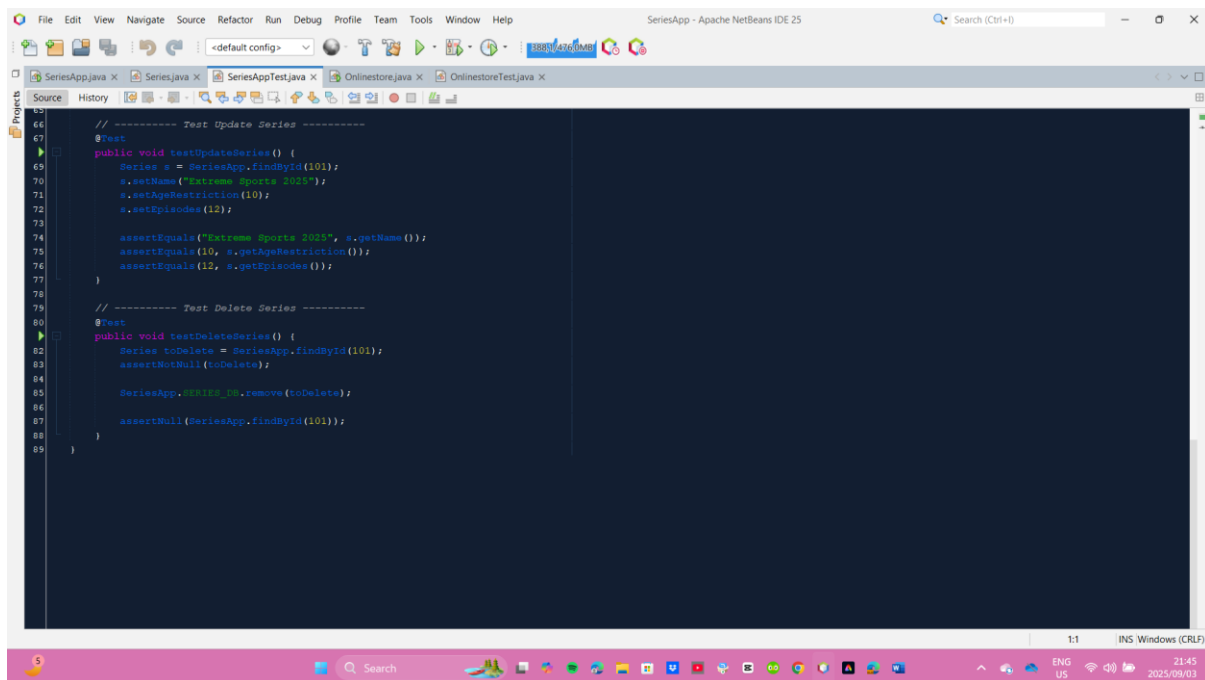
## SECTION B: UNIT TESTING



```
1 //SECTION B :UNIT TEST
2 package seriesapp;
3
4 import org.junit.Before;
5 import org.junit.Test;
6 import static org.junit.Assert.*;
7
8 public class SeriesAppTest {
9
10     @Before
11     public void setUp() {
12         // Clear database before each test
13         SeriesApp.seriesDB.clear();
14         // Add one sample series
15         SeriesApp.seriesDB.add(new Series(101, "Extreme Sports", 12, 10));
16     }
17
18     // ----- Test isInteger -----
19     @Test
20     public void testIsInteger_ValidNumber() {
21         assertTrue(SeriesApp.isInteger("123"));
22     }
23
24     @Test
25     public void testIsInteger_InvalidNumber() {
26         assertFalse(SeriesApp.isInteger("abc"));
27     }
28
29     @Test
30     public void testIsInteger_EmptyString() {
31         assertFalse(SeriesApp.isInteger(""));
32     }
33
34     @Test
35     public void testIsInteger_Null() {
36         assertFalse(SeriesApp.isInteger(null));
37     }
38 }
```



```
37
38
39     // ----- Test findById -----
40     @Test
41     public void testFindById_Found() {
42         Series s = SeriesApp.findById(101);
43         assertNotNull(s);
44         assertEquals("Extreme Sports", s.getName());
45         assertEquals(12, s.getAgeRestriction());
46         assertEquals(10, s.getEpisodes());
47     }
48
49     @Test
50     public void testFindById_NotFound() {
51         Series s = SeriesApp.findById(999);
52         assertNull(s);
53     }
54
55     // ----- Test Add Series -----
56     @Test
57     public void testAddSeries() {
58         Series newSeries = new Series(102, "Bargain Hunters", 10, 8);
59         SeriesApp.seriesDB.add(newSeries);
60
61         Series found = SeriesApp.findById(102);
62         assertNotNull(found);
63         assertEquals("Bargain Hunters", found.getName());
64     }
65
66     // ----- Test Update Series -----
67     @Test
68     public void testUpdateSeries() {
69         Series s = SeriesApp.findById(101);
70         s.setName("Extreme Sports 2025");
71         s.setAgeRestriction(10);
72         s.setEpisodes(12);
73     }
74 }
```



The screenshot shows the Apache NetBeans IDE with the file `SeriesAppTest.java` open. The code contains two test methods: `testUpdateSeries()` and `testDeleteSeries()`. The `testUpdateSeries()` method creates a `Series` object with ID 101, sets its name to "Extreme Sports 2025", age restriction to 10, and episodes to 12. It then asserts that the object's name, age restriction, and episodes match the expected values. The `testDeleteSeries()` method finds the series with ID 101, removes it from the `SeriesApp` list, and asserts that the series is now null.

```
// ----- Test Update Series -----
@Test
public void testUpdateSeries() {
    Series s = SeriesApp.findById(101);
    s.setName("Extreme Sports 2025");
    s.setAgeRestriction(10);
    s.setEpisodes(12);

    assertEquals("Extreme Sports 2025", s.getName());
    assertEquals(10, s.getAgeRestriction());
    assertEquals(12, s.getEpisodes());
}

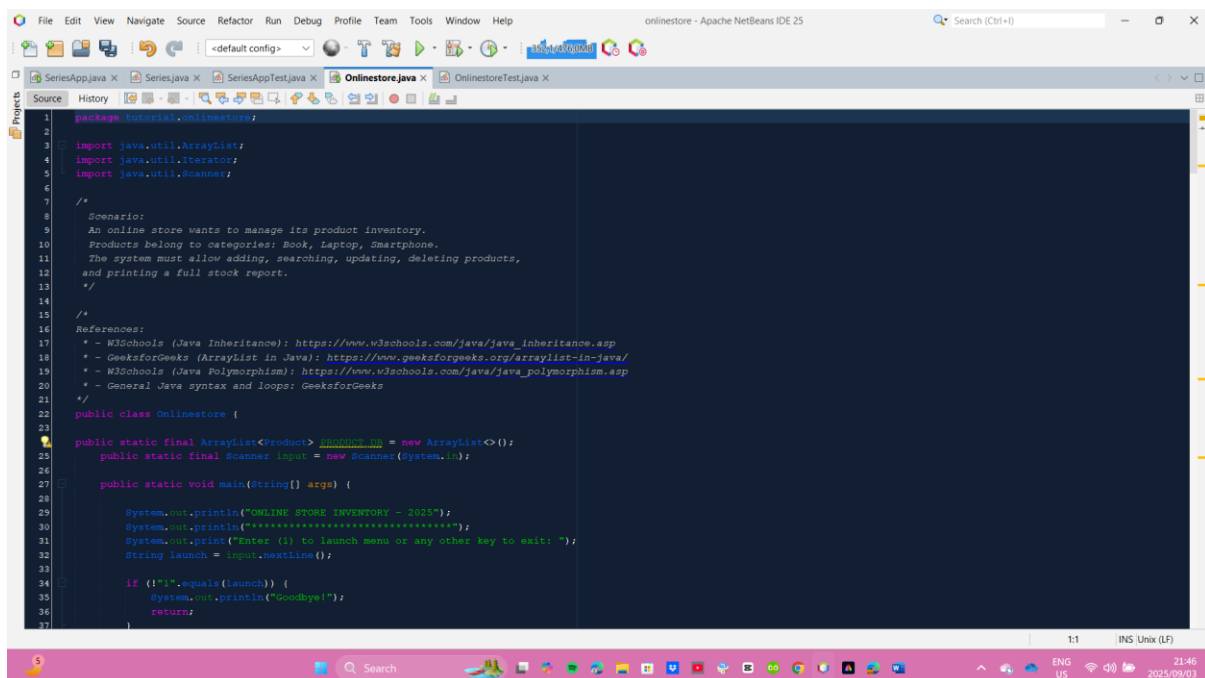
// ----- Test Delete Series -----
@Test
public void testDeleteSeries() {
    Series toDelete = SeriesApp.findById(101);
    assertNotNull(toDelete);

    SeriesApp.getSeries().remove(toDelete);
    assertNull(SeriesApp.findById(101));
}
```

## SECTION B : OWN APP

### Scenario:

An online store wants to manage its product inventory. Products belong to categories: Book, Laptop, Smartphone. The system must allow adding, searching, updating, deleting products, and printing a full stock report. MAIN CLASS: `OnlineStore.java`



The screenshot shows the Apache NetBeans IDE with the file `OnlineStore.java` open. The code defines a `OnlineStore` class that manages a list of products. It includes imports for `ArrayList`, `Iterator`, and `Scanner`. The class has a static `ArrayList<Product>` named `products` and a static `Scanner` named `input`. The `main` method prints the online store inventory for 2025, prompts the user to launch a menu or press any key to exit, and then prints a goodbye message.

```
package tutorial.onlinestore;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.Scanner;

/*
 * Scenario:
 * An online store wants to manage its product inventory.
 * Products belong to categories: Book, Laptop, Smartphone.
 * The system must allow adding, searching, updating, deleting products,
 * and printing a full stock report.
 */

/*
 * References:
 * - W3Schools (Java Inheritance): https://www.w3schools.com/java/java_inheritance.asp
 * - GeeksforGeeks (ArrayList in Java): https://www.geeksforgeeks.org/arraylist-in-java/
 * - W3Schools (Java Polymorphism): https://www.w3schools.com/java/java_polymorphism.asp
 * - General Java syntax and loops: GeeksforGeeks
 */

public class OnlineStore {

    public static final ArrayList<Product> products = new ArrayList<>();
    public static final Scanner input = new Scanner(System.in);

    public static void main(String[] args) {

        System.out.println("ONLINE STORE INVENTORY - 2025");
        System.out.println("-----");
        System.out.print("Enter (1) to launch menu or any other key to exit: ");
        String launch = input.nextLine();

        if (!"1".equals(launch)) {
            System.out.println("Goodbye!");
            return;
        }
    }
}
```

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help onlinestore - Apache NetBeans IDE 25 Search (Ctrl+F)
<default config>
SeriesApp.java X SeriesAppTest.java X OnlineStore.java X OnlineStoreTest.java X
Source History
27
38
39 boolean running = true;
40 while (running) {
41     printMenu();
42     String choice = input.nextLine();
43
44     switch (choice) {
45         case "1" -> addProduct();
46         case "2" -> searchProduct();
47         case "3" -> updateProduct();
48         case "4" -> deleteProduct();
49         case "5" -> printReport();
50         case "6" -> {
51             System.out.println("Exiting application...");
52             running = false;
53         }
54         default -> System.out.println("Invalid option! Please choose 1-6.");
55     }
56
57     if (running) {
58         System.out.print("\nEnter (l) to return to menu or any other key to exit: ");
59         String again = input.nextLine();
60         if (!"l".equals(again)) {
61             System.out.println("Goodbye!");
62             break;
63         }
64     }
65 }
66
67 // Menu
68 public static void printMenu() {
69     System.out.println("Welcome! select one of the following menu items:");
70     System.out.println("(1) Add a new product");
71     System.out.println("(2) Search for a product");
72     System.out.println("(3) Update product details");
73     System.out.println("(4) Delete a product");
74     System.out.println("(5) Print inventory report");
75     System.out.println("(6) Exit application");
76     System.out.print("Your choice: ");
77 }
78
79 // 1. Add Product
80 public static void addProduct() {
81     System.out.println("\nADD A NEW PRODUCT");
82     System.out.println("*****");
83
84     System.out.print("Enter Product ID: ");
85     int id = Integer.parseInt(input.nextLine());
86
87     System.out.print("Enter Product Name: ");
88     String name = input.nextLine();
89
90     System.out.println("Select Product Type: (1) Book, (2) Laptop, (3) Smartphone");
91     String type = input.nextLine();
92
93     Product newProduct = null;
94     switch (type) {
95         case "1" -> {
96             System.out.print("Enter Author Name: ");
97             String author = input.nextLine();
98             newProduct = new Book(id, name, author);
99         }
100         case "2" -> {
101             System.out.print("Enter RAM size (GB): ");
102             int ram = Integer.parseInt(input.nextLine());
103             newProduct = new Laptop(id, name, ram);
104         }
105         case "3" -> {
106             System.out.print("Enter Camera (MP): ");
107             int cam = Integer.parseInt(input.nextLine());
108             newProduct = new Smartphone(id, name, cam);
109         }
110     }
111 }
```

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help onlinestore - Apache NetBeans IDE 25 Search (Ctrl+F)
<default config>
SeriesApp.java X SeriesAppTest.java X OnlineStore.java X OnlineStoreTest.java X
Source History
73
74 System.out.println("(3) Update product details");
75 System.out.println("(4) Delete a product");
76 System.out.println("(5) Print inventory report");
77 System.out.println("(6) Exit application");
78 System.out.print("Your choice: ");
79 }
80
81 // 1. Add Product
82 public static void addProduct() {
83     System.out.println("\nADD A NEW PRODUCT");
84     System.out.println("*****");
85
86     System.out.print("Enter Product ID: ");
87     int id = Integer.parseInt(input.nextLine());
88
89     System.out.print("Enter Product Name: ");
90     String name = input.nextLine();
91
92     System.out.println("Select Product Type: (1) Book, (2) Laptop, (3) Smartphone");
93     String type = input.nextLine();
94
95     Product newProduct = null;
96     switch (type) {
97         case "1" -> {
98             System.out.print("Enter Author Name: ");
99             String author = input.nextLine();
100             newProduct = new Book(id, name, author);
101         }
102         case "2" -> {
103             System.out.print("Enter RAM size (GB): ");
104             int ram = Integer.parseInt(input.nextLine());
105             newProduct = new Laptop(id, name, ram);
106         }
107         case "3" -> {
108             System.out.print("Enter Camera (MP): ");
109             int cam = Integer.parseInt(input.nextLine());
110             newProduct = new Smartphone(id, name, cam);
111         }
112     }
113 }
```

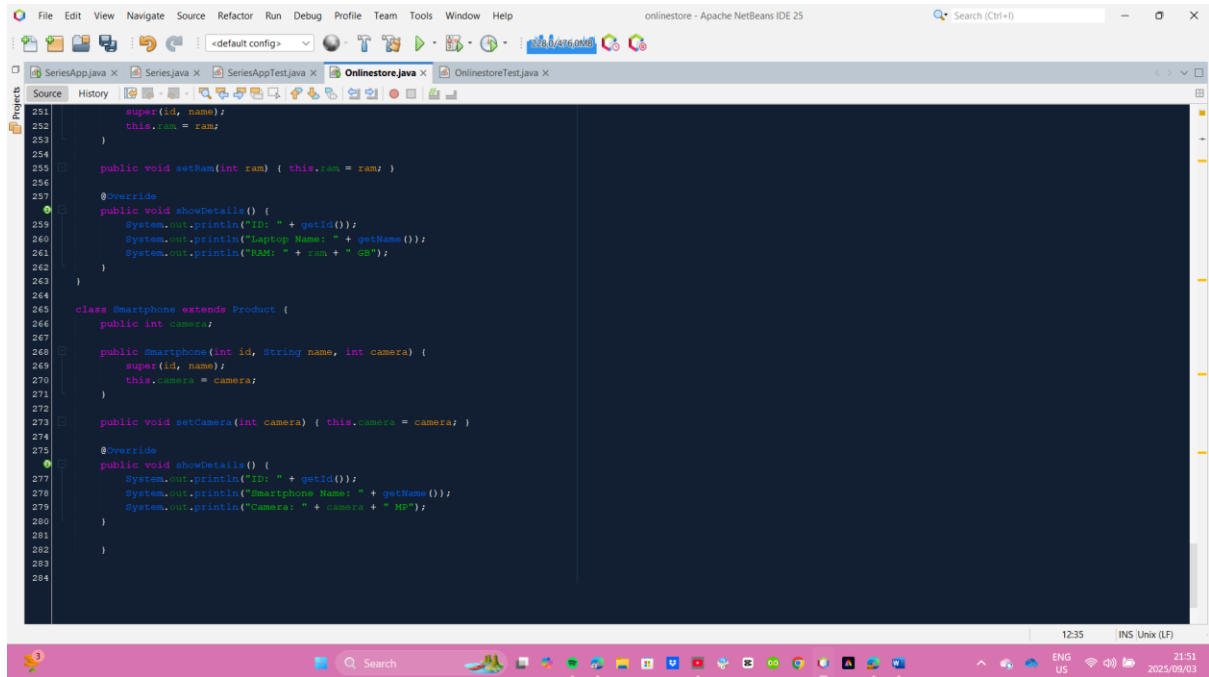
```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help onlinestore - Apache NetBeans IDE 25
<default config> 423.0/426.0MB
SeriesApp.java x SeriesAppTest.java x Onlinestore.java x OnlinestoreTest.java x
Source History
109         newProduct = new Smartphone(id, name, cam);
110     }
111     default -> {
112         System.out.println("Invalid type! Product not added.");
113         return;
114     }
115 }
116
117 PRODUCT_DB.add(newProduct);
118 System.out.println("Product added successfully!");
119 }
120
121 // 2. Search Product
122 public static void searchProduct() {
123     System.out.print("\nEnter Product ID to search: ");
124     int id = Integer.parseInt(input.nextLine());
125     Product p = findById(id);
126
127     if (p == null) {
128         System.out.println("Product with ID " + id + " was not found!");
129     } else {
130         System.out.println("Product found!");
131         p.showDetails(); // Polymorphism in action
132     }
133 }
134
135 // 3. Update Product
136 public static void updateProduct() {
137     System.out.print("\nEnter Product ID to update: ");
138     int id = Integer.parseInt(input.nextLine());
139     Product p = findById(id);
140
141     if (p == null) {
142         System.out.println("Product with ID " + id + " was not found!");
143         return;
144     }
145 }
```

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help onlinestore - Apache NetBeans IDE 25
<default config> 334.6/426.0MB
SeriesApp.java x SeriesAppTest.java x Onlinestore.java x OnlinestoreTest.java x
Source History
144 }
145
146 System.out.print("Enter new Product Name: ");
147 String newName = input.nextLine();
148 p.setName(newName);
149
150 if (p instanceof Book b) {
151     System.out.print("Enter new Author: ");
152     b.setAuthor(input.nextLine());
153 } else if (p instanceof Laptop l) {
154     System.out.print("Enter new RAM size: ");
155     l.setRam(Integer.parseInt(input.nextLine()));
156 } else if (p instanceof Smartphone s) {
157     System.out.print("Enter new Camera MP: ");
158     s.setCamera(Integer.parseInt(input.nextLine()));
159 }
160
161 System.out.println("Update completed!");
162 }
163
164 // 4. Delete Product
165 public static void deleteProduct() {
166     System.out.print("\nEnter Product ID to delete: ");
167     int id = Integer.parseInt(input.nextLine());
168
169     Iterator<Product> it = PRODUCT_DB.iterator();
170     while (it.hasNext()) {
171         if (it.next().getId() == id) {
172             it.remove();
173             System.out.println("Product with ID " + id + " was deleted!");
174             return;
175         }
176     }
177     System.out.println("Product with ID " + id + " not found!");
178 }
179
180 // 5. Print Report
```



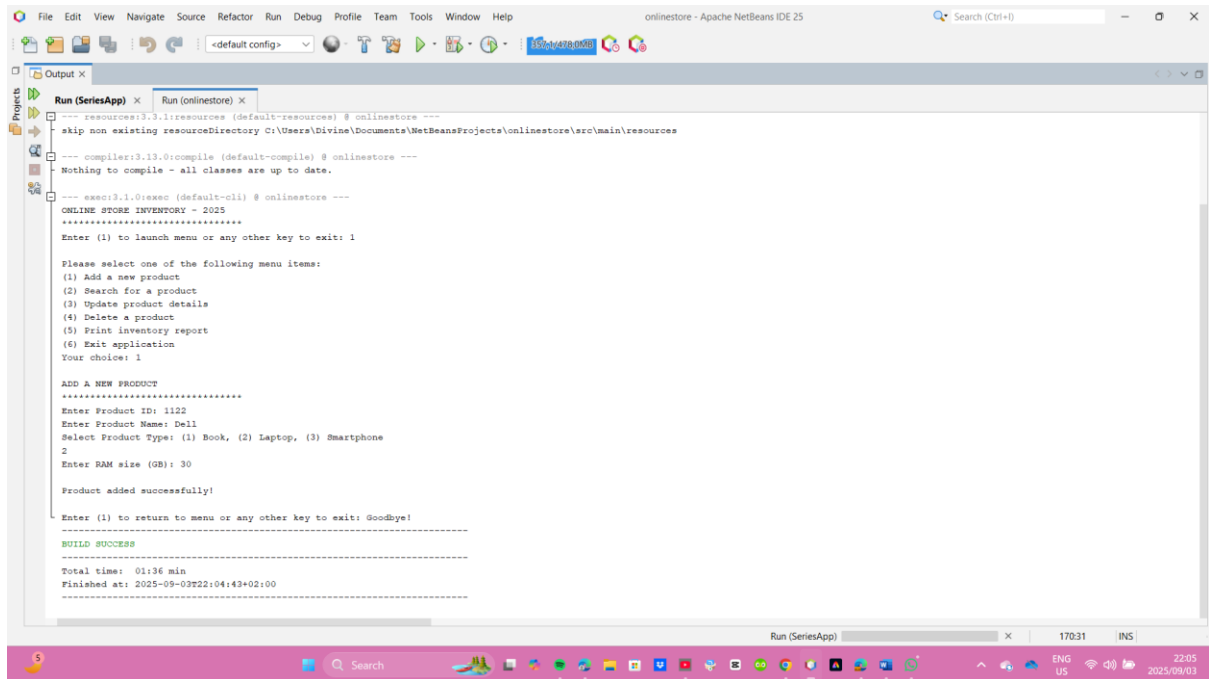
```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help onlinestore - Apache NetBeans IDE 25
<default config> 138.0/276.0MB
SeriesApp.java x SeriesAppTest.java x OnlinestoreTest.java x OnlinestoreTest.java x
Source History
180 // 5. Print Report
181 public static void printReport() {
182     if (inventory.isEmpty()) {
183         System.out.println("No products in inventory.");
184         return;
185     }
186     System.out.println("\nINVENTORY REPORT:");
187     System.out.println("*****");
188     for (int i = 0; i < PRODUCT_DB.size(); i++) {
189         System.out.println("Product " + (i + 1));
190         inventory.get(i).showDetails(); // Polymorphism
191         System.out.println("-----");
192     }
193 }
194
195 // Helper: Find product by ID
196 public static Product findById(int id) {
197     for (Product p : PRODUCT_DB) {
198         if (p.getId() == id) return p;
199     }
200     return null;
201 }
202
203 public static void addProductDirect(java.awt.print.Book b) {
204     throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody
205 }
206
207 public ArrayList<Object> products;
208
209 /* =====
210 Base Class + Subclasses
211 ===== */
212
213 abstract class Product {
214     public int id;
215     public String name;
```

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help onlinestore - Apache NetBeans IDE 25
<default config> 138.0/276.0MB
SeriesApp.java x SeriesAppTest.java x OnlinestoreTest.java x OnlinestoreTest.java x
Source History
215     public String name;
216
217     public Product(int id, String name) {
218         this.id = id;
219         this.name = name;
220     }
221
222     public int getId() { return id; }
223     public String getName() { return name; }
224     public void setName(String name) { this.name = name; }
225
226     public abstract void showDetails(); // Overridden by subclasses
227 }
228
229 class Book extends Product {
230     public String author;
231
232     public Book(int id, String name, String author) {
233         super(id, name);
234         this.author = author;
235     }
236
237     public void setAuthor(String author) { this.author = author; }
238
239     @Override
240     public void showDetails() {
241         System.out.println("ID: " + getId());
242         System.out.println("Book Name: " + getName());
243         System.out.println("Author: " + author);
244     }
245 }
246
247 class Laptop extends Product {
248     public int ram;
249
250     public Laptop(int id, String name, int ram) {
251         super(id, name);
```



```
251     super(id, name);
252     this.ram = ram;
253 }
254
255 public void setRam(int ram) { this.ram = ram; }
256
257 @Override
258 public void showDetails() {
259     System.out.println("ID: " + getId());
260     System.out.println("Laptop Name: " + getName());
261     System.out.println("RAM: " + ram + " GB");
262 }
263 }
264
265 class Smartphone extends Product {
266     public int camera;
267
268     public Smartphone(int id, String name, int camera) {
269         super(id, name);
270         this.camera = camera;
271     }
272
273     public void setCamera(int camera) { this.camera = camera; }
274
275     @Override
276     public void showDetails() {
277         System.out.println("ID: " + getId());
278         System.out.println("Smartphone Name: " + getName());
279         System.out.println("Camera: " + camera + " MP");
280     }
281 }
282
283
284
```

## OUTPUT:



```
Run (SeriesApp) X Run (onlinestore) X
--- resources:3.3.1:resources (default-resources) @ onlinestore ---
skip non existing resourceDirectory C:\Users\Divine\Documents\NetBeansProjects\onlinestore\src\main\resources

--- compiler:3.13.0:compile (default-compile) @ onlinestore ---
Nothing to compile - all classes are up to date.

--- exec:3.1.0:exec (default-cli) @ onlinestore ---
ONLINE STORE INVENTORY - 2025
*****
Enter (1) to launch menu or any other key to exit: 1

Please select one of the following menu items:
(1) Add a new product
(2) Search for a product
(3) Update product details
(4) Delete a product
(5) Print inventory report
(6) Exit application
Your choice: 1

ADD A NEW PRODUCT
*****
Enter Product ID: 1122
Enter Product Name: Dell
Select Product Type: (1) Book, (2) Laptop, (3) Smartphone
2
Enter RAM size (GB): 30

Product added successfully!

Enter (1) to return to menu or any other key to exit: Goodbye!

BUILD SUCCESS

Total time: 01:36 min
Finished at: 2025-09-03T22:04:43+02:00
-----
```

## CODE:

### MAIN CLASS

```
package tutorial.onlinestore;
```

```
import java.util.ArrayList;
```

```
import java.util.Iterator;
```

```
import java.util.Scanner;
```

```
/*
```

Scenario:

An online store wants to manage its product inventory.

Products belong to categories: Book, Laptop, Smartphone.

The system must allow adding, searching, updating, deleting products,  
and printing a full stock report.

```
*/
```

```
/*
```

References:

\* - W3Schools (Java Inheritance):

[https://www.w3schools.com/java/java\\_inheritance.asp](https://www.w3schools.com/java/java_inheritance.asp)

\* - GeeksforGeeks (ArrayList in Java): <https://www.geeksforgeeks.org/arraylist-in-java/>

\* - W3Schools (Java Polymorphism):

[https://www.w3schools.com/java/java\\_polymorphism.asp](https://www.w3schools.com/java/java_polymorphism.asp)

\* - General Java syntax and loops: GeeksforGeeks

```
*/
```

```
public class Onlinestore {
```

```
public static final ArrayList<Product> PRODUCT_DB = new ArrayList<>();

public static final Scanner input = new Scanner(System.in);

public static void main(String[] args) {

    System.out.println("ONLINE STORE INVENTORY - 2025");

    System.out.println("*****");

    System.out.print("Enter (1) to launch menu or any other key to exit: ");

    String launch = input.nextLine();

    if (!"1".equals(launch)) {

        System.out.println("Goodbye!");

        return;

    }

    boolean running = true;

    while (running) {

        printMenu();

        String choice = input.nextLine();

        switch (choice) {

            case "1" -> addProduct();

            case "2" -> searchProduct();

            case "3" -> updateProduct();

            case "4" -> deleteProduct();

            case "5" -> printReport();

            case "6" -> {

                System.out.println("Exiting application...");
```

```

        running = false;
    }
    default -> System.out.println("Invalid option! Please choose 1-6.");
}

if (running) {
    System.out.print("\nEnter (1) to return to menu or any other key to exit: ");
    String again = input.nextLine();
    if (!"1".equals(again)) {
        System.out.println("Goodbye!");
        break;
    }
}
}
}

// Menu
public static void printMenu() {
    System.out.println("\nPlease select one of the following menu items:");
    System.out.println("(1) Add a new product");
    System.out.println("(2) Search for a product");
    System.out.println("(3) Update product details");
    System.out.println("(4) Delete a product");
    System.out.println("(5) Print inventory report");
    System.out.println("(6) Exit application");
    System.out.print("Your choice: ");
}

```

```
// 1. Add Product

public static void addProduct() {

    System.out.println("\nADD A NEW PRODUCT");

    System.out.println("*****");

    System.out.print("Enter Product ID: ");

    int id = Integer.parseInt(input.nextLine());

    System.out.print("Enter Product Name: ");

    String name = input.nextLine();

    System.out.println("Select Product Type: (1) Book, (2) Laptop, (3) Smartphone");

    String type = input.nextLine();

    Product newProduct = null;

    switch (type) {

        case "1" -> {

            System.out.print("Enter Author Name: ");

            String author = input.nextLine();

            newProduct = new Book(id, name, author);

        }

        case "2" -> {

            System.out.print("Enter RAM size (GB): ");

            int ram = Integer.parseInt(input.nextLine());

            newProduct = new Laptop(id, name, ram);

        }

        case "3" -> {

            System.out.print("Enter Camera (MP): ");
```

```

        int cam = Integer.parseInt(input.nextLine());

        newProduct = new Smartphone(id, name, cam);
    }

    default -> {

        System.out.println("Invalid type! Product not added.");

        return;

    }

}

PRODUCT_DB.add(newProduct);

System.out.println("Product added successfully!");

}

// 2. Search Product

public static void searchProduct() {

    System.out.print("\nEnter Product ID to search: ");

    int id = Integer.parseInt(input.nextLine());

    Product p = findById(id);

    if (p == null) {

        System.out.println("Product with ID " + id + " was not found!");

    } else {

        System.out.println("Product found:");

        p.showDetails(); // Polymorphism in action

    }

}

// 3. Update Product

```

```
public static void updateProduct() {  
    System.out.print("\nEnter Product ID to update: ");  
    int id = Integer.parseInt(input.nextLine());  
    Product p = findById(id);  
  
    if (p == null) {  
        System.out.println("Product with ID " + id + " was not found!");  
        return;  
    }  
  
    System.out.print("Enter new Product Name: ");  
    String newName = input.nextLine();  
    p.setName(newName);  
  
    if (p instanceof Book b) {  
        System.out.print("Enter new Author: ");  
        b.setAuthor(input.nextLine());  
    } else if (p instanceof Laptop l) {  
        System.out.print("Enter new RAM size: ");  
        l.setRam(Integer.parseInt(input.nextLine()));  
    } else if (p instanceof Smartphone s) {  
        System.out.print("Enter new Camera MP: ");  
        s.setCamera(Integer.parseInt(input.nextLine()));  
    }  
  
    System.out.println("Update completed!");  
}
```



// 4. Delete Product

```
public static void deleteProduct() {  
    System.out.print("\nEnter Product ID to delete: ");  
    int id = Integer.parseInt(input.nextLine());  
  
    Iterator<Product> it = PRODUCT_DB.iterator();  
    while (it.hasNext()) {  
        if (it.next().getId() == id) {  
            it.remove();  
            System.out.println("Product with ID " + id + " was deleted!");  
            return;  
        }  
    }  
    System.out.println("Product with ID " + id + " not found!");  
}
```

// 5. Print Report

```
public static void printReport() {  
    if (PRODUCT_DB.isEmpty()) {  
        System.out.println("No products in inventory.");  
        return;  
    }  
    System.out.println("\nINVENTORY REPORT");  
    System.out.println("*****");  
    for (int i = 0; i < PRODUCT_DB.size(); i++) {  
        System.out.println("Product " + (i + 1));  
        PRODUCT_DB.get(i).showDetails(); // Polymorphism  
        System.out.println("-----");  
    }  
}
```

```
    }  
}
```

// Helper: Find product by ID

```
public static Product findById(int id) {  
    for (Product p : PRODUCT_DB) {  
        if (p.getId() == id) return p;  
    }  
    return null;  
}
```

```
public static void addProductDirect(java.awt.print.Book b) {  
    throw new UnsupportedOperationException("Not supported yet."); // Generated  
from  
nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody  
}  
public ArrayList<Object> products;  
}
```

```
/* =====
```

Base Class + Subclasses

```
===== */
```

```
abstract class Product {  
    public int id;  
    public String name;  
  
    public Product(int id, String name) {  
        this.id = id;
```

```
        this.name = name;
    }

    public int getId() { return id; }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public abstract void showDetails(); // Overridden by subclasses
}

class Book extends Product {

    public String author;

    public Book(int id, String name, String author) {

        super(id, name);

        this.author = author;

    }

    public void setAuthor(String author) { this.author = author; }

    @Override
    public void showDetails() {

        System.out.println("ID: " + getId());

        System.out.println("Book Name: " + getName());

        System.out.println("Author: " + author);

    }
}
```

```
class Laptop extends Product {  
    public int ram;  
  
    public Laptop(int id, String name, int ram) {  
        super(id, name);  
        this.ram = ram;  
    }  
  
    public void setRam(int ram) { this.ram = ram; }  
  
    @Override  
    public void showDetails() {  
        System.out.println("ID: " + getId());  
        System.out.println("Laptop Name: " + getName());  
        System.out.println("RAM: " + ram + " GB");  
    }  
}  
  
class Smartphone extends Product {  
    public int camera;  
  
    public Smartphone(int id, String name, int camera) {  
        super(id, name);  
        this.camera = camera;  
    }  
  
    public void setCamera(int camera) { this.camera = camera; }
```

@Override

public void showDetails() {

    System.out.println("ID: " + getId());

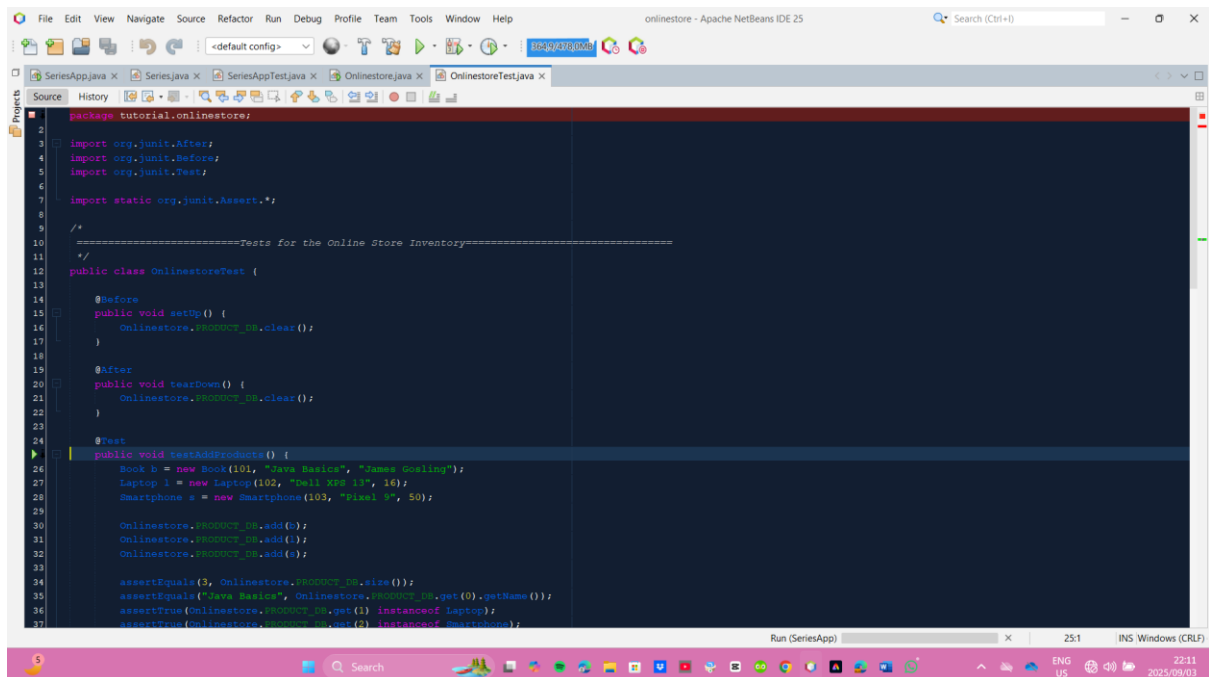
    System.out.println("Smartphone Name: " + getName());

    System.out.println("Camera: " + camera + " MP");

}

}

# UNIT TESTING:



```
package tutorial.onlinestore;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import static org.junit.Assert.*;

/*
=====Tests for the Online Store Inventory=====
*/
public class OnlineStoreTest {

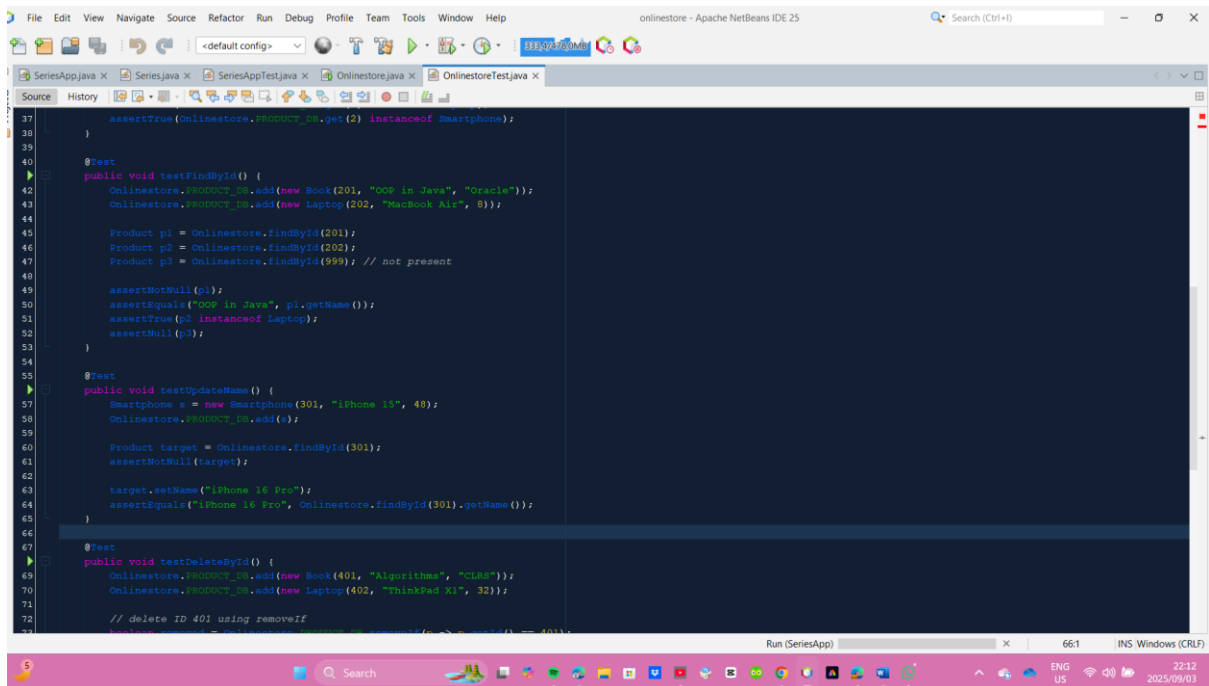
    @Before
    public void setUp() {
        OnlineStore.PRODUCT_DB.clear();
    }

    @After
    public void tearDown() {
        OnlineStore.PRODUCT_DB.clear();
    }

    @Test
    public void testAddProducts() {
        Book b = new Book(101, "Java Basics", "James Gosling");
        Laptop l = new Laptop(102, "Dell XPS 13", 16);
        Smartphone s = new Smartphone(103, "Pixel 6", 50);

        OnlineStore.PRODUCT_DB.add(b);
        OnlineStore.PRODUCT_DB.add(l);
        OnlineStore.PRODUCT_DB.add(s);

        assertEquals(3, OnlineStore.PRODUCT_DB.size());
        assertEquals("Java Basics", OnlineStore.PRODUCT_DB.get(0).getName());
        assertTrue(OnlineStore.PRODUCT_DB.get(1) instanceof Laptop);
        assertTrue(OnlineStore.PRODUCT_DB.get(2) instanceof Smartphone);
    }
}
```



```
        assertTrue(OnlineStore.PRODUCT_DB.get(2) instanceof Smartphone);
    }

    @Test
    public void testFindById() {
        OnlineStore.PRODUCT_DB.add(new Book(201, "OOP in Java", "Opacile"));
        OnlineStore.PRODUCT_DB.add(new Laptop(202, "MacBook Air", 8));

        Product p1 = OnlineStore.findById(201);
        Product p2 = OnlineStore.findById(202);
        Product p3 = OnlineStore.findById(999); // not present

        assertNotNull(p1);
        assertEquals("OOP in Java", p1.getName());
        assertTrue(p2 instanceof Laptop);
        assertNull(p3);
    }

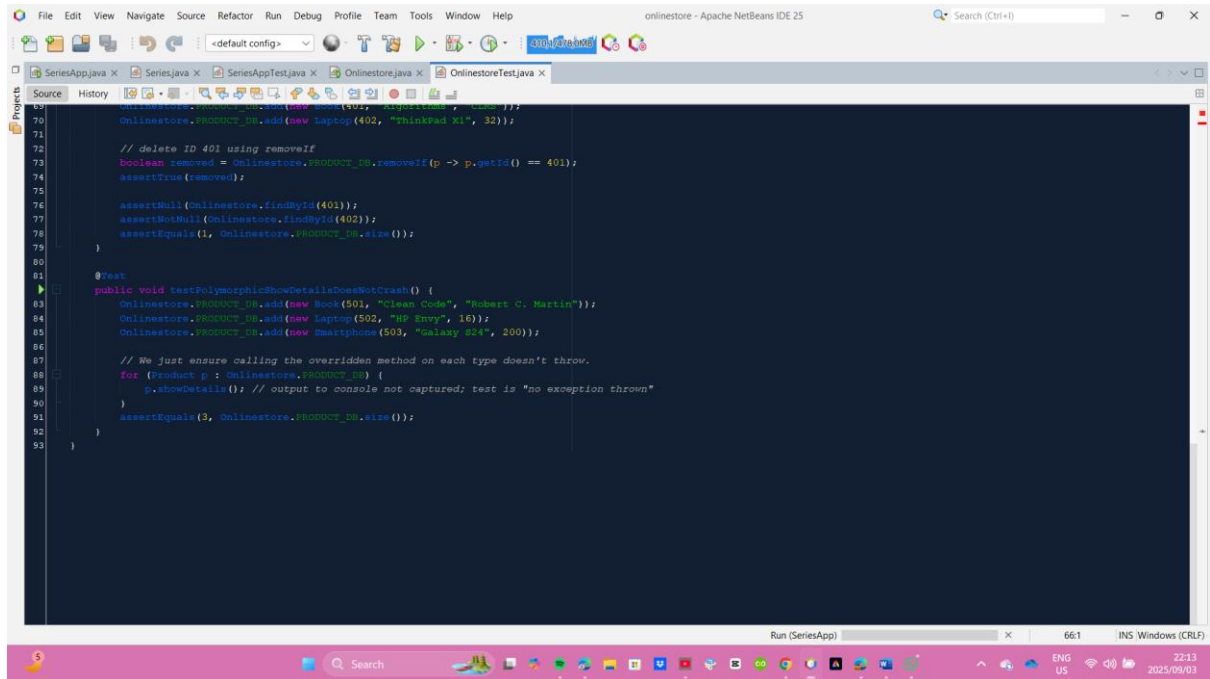
    @Test
    public void testUpdateName() {
        Smartphone s = new Smartphone(301, "iPhone 13", 48);
        OnlineStore.PRODUCT_DB.add(s);

        Product target = OnlineStore.findById(301);
        assertNotNull(target);

        target.setName("iPhone 16 Pro");
        assertEquals("iPhone 16 Pro", OnlineStore.findById(301).getName());
    }

    @Test
    public void testDeleteById() {
        OnlineStore.PRODUCT_DB.add(new Book(401, "Algorithms", "CLR"));
        OnlineStore.PRODUCT_DB.add(new Laptop(402, "ThinkPad X1", 32));

        // delete ID 401 using removeIf
        OnlineStore.PRODUCT_DB.removeIf(p -> p.getId() == 401);
    }
}
```



## OUTPUT:

