**WEEK 1 HANDS ON EXERCISE – DATA STRUCTURES AND ALGORITHMS**

**Exercise 1: Inventory Management System**

**PROBLEM STATEMENT:**

You are developing an inventory management system for a warehouse. Efficient data storage and retrieval are crucial.

**CODE:**

**Product.java**

package InventoryManagementSystem;

public class Product

{

private int productId;

private String productName;

private int quantity;

private double price;

public Product(int productId,String productName,int quantity, double price)

{

this.productId = productId;

this.productName = productName;

this.quantity = quantity;

this.price = price;

}

// getters

public int getId()

{

return productId;

}

public String getName()

{

return productName;

}

public int getQty()

{

return quantity;

}

public double getprice()

{

return price;

}

// setters

public void setQty(int quantity)

{

this.quantity = quantity;

}

public void setprice(double price)

{

this.price = price;

}

*@Override*

public String toString()

{

return "\nID: " + productId + ", Name: " + productName + ", Qty: " + quantity + ", Price: $" + price + "\n";

}

}

**Inventory.java**

package InventoryManagementSystem;

import java.util.\*;

public class Inventory

{

// Key = product ID (Integer), Value = Product object

private Map<Integer, Product> map = new HashMap<>();

// add

public void addProduct(Product product)

{

if (map.containsKey(product.getId()))

{

System.***out***.println("\nProduct updated: ID " + product.getId());

}

else

{

System.***out***.println("\nProduct added: ID " + product.getId());

}

map.put(product.getId(), product);

// 101-key , Product object for "Mouse" - value

}

// update

public void updateProduct(int productId, int quantity, double price)

{

Product p = map.get(productId);

if(p != null)

{

p.setQty(quantity);

p.setprice(price);

System.***out***.println("\nProduct updated: ID " + productId);

}

else

{

System.***out***.println("\nProduct not found: ID " + productId);

}

}

// delete

public void deleteProduct(int productId)

{

if (map.containsKey(productId))

{

map.remove(productId);

System.***out***.println("\nProduct deleted: ID " + productId);

}

else

{

System.***out***.println("\nProduct not found: ID " + productId);

}

}

public void display()

{

// get all Product objects

for(Product p : map.values())

{

// This will call the Product class's toString() method

System.***out***.print(p);

}

}

}

**Main.java**

package InventoryManagementSystem;

import java.util.\*;

public class Main

{

public static void main(String args[])

{

Inventory manager = new Inventory();

Product p1 = new Product(101, "Laptop", 2, 55000.0);

Product p2 = new Product(102, "Mouse", 10, 499.99);

Product p3 = new Product(103, "Keyboard", 5, 999.50);

Product p4 = new Product(104, "Monitor", 3, 12000.0);

manager.addProduct(p1);

manager.addProduct(p2);

manager.addProduct(p3);

manager.addProduct(p4);

manager.display();

manager.updateProduct(101, 5, 53000.0);

manager.deleteProduct(102);

manager.deleteProduct(103);

manager.updateProduct(104, 3, 11000.0);

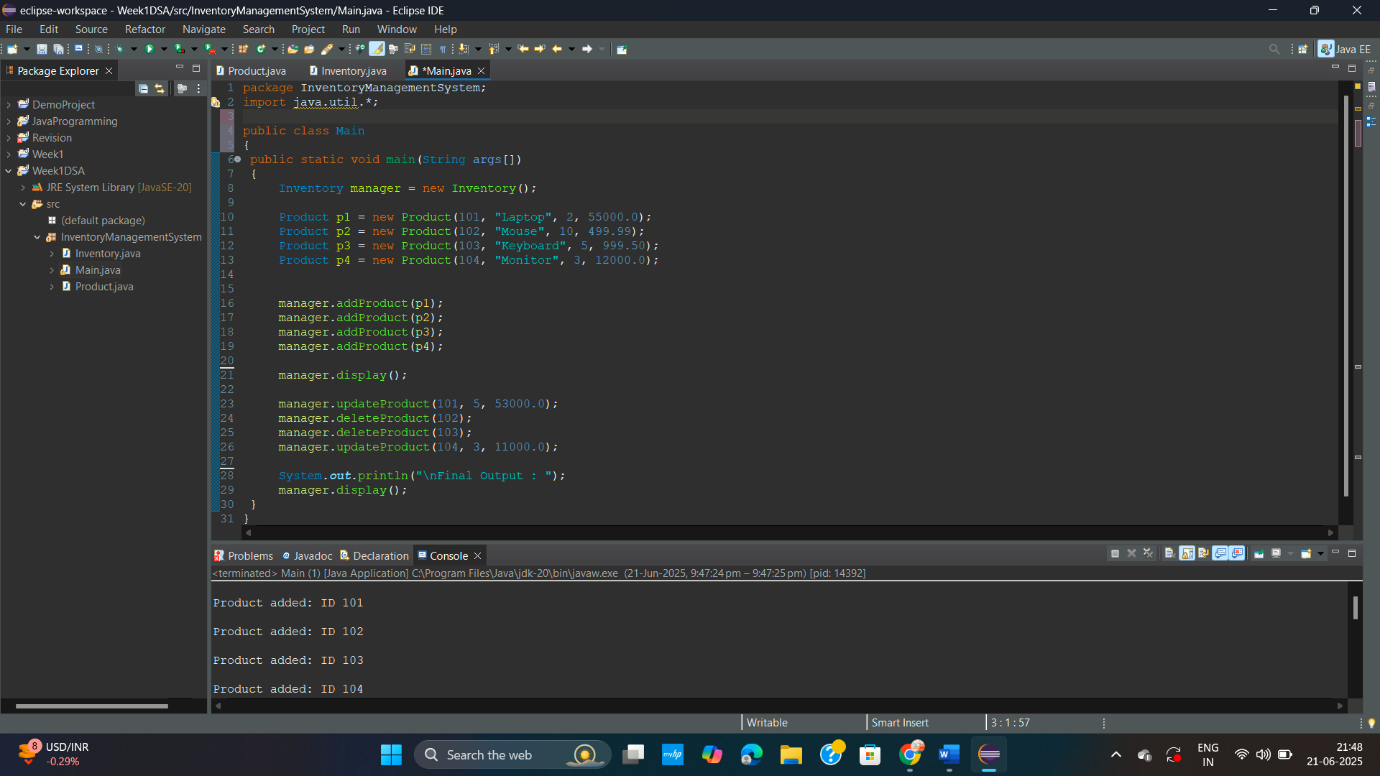
System.***out***.println("\nFinal Output : ");

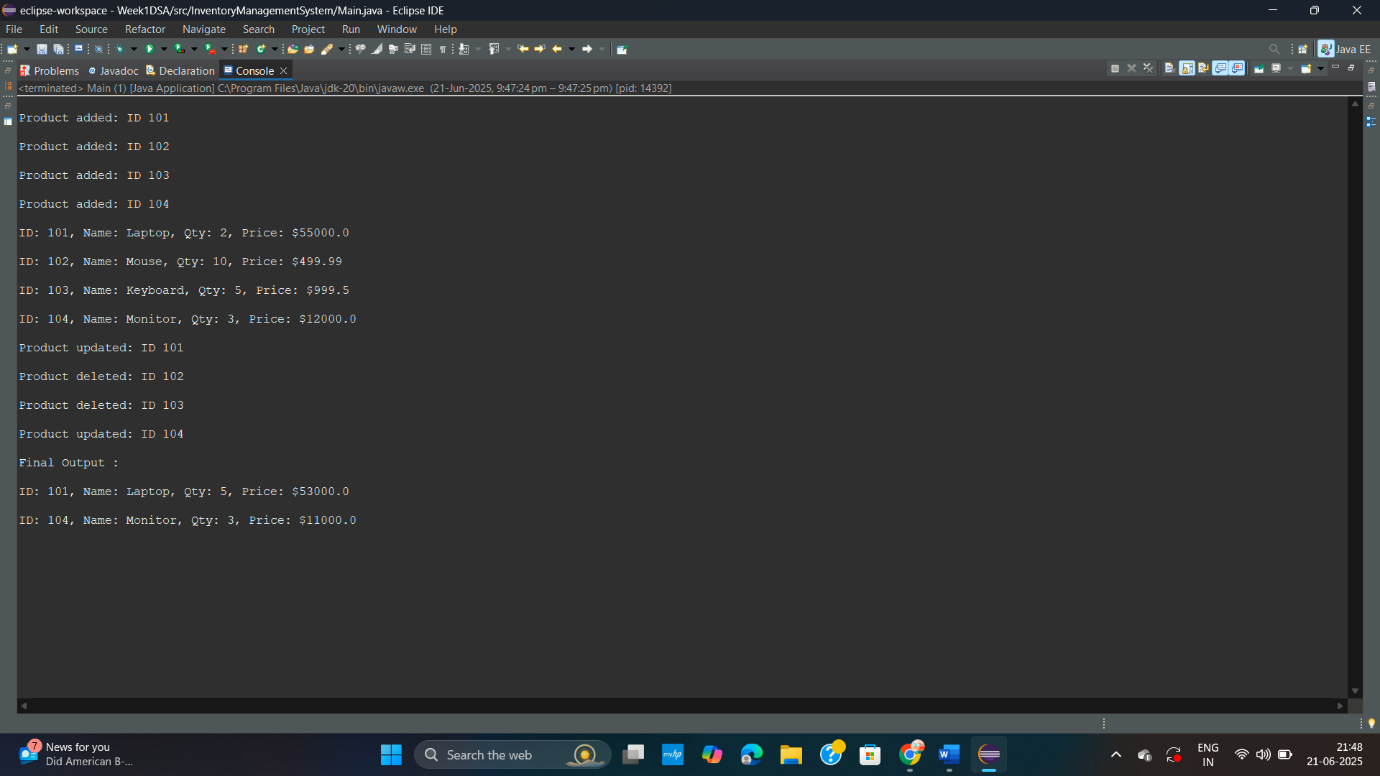
manager.display();

}

}

**OUTPUT :**





**Exercise 2: E-commerce Platform Search Function**

**PROBLEM STATEMENT:**

You are working on the search functionality of an e-commerce platform. The search needs to be optimized for fast performance.

**CODE:**

**Product.java**

package ECommerceSearchFunctions;

public class Product

{

private int productId;

private String productName;

private String category;

public Product(int productId,String productName,String category)

{

this.productId = productId;

this.productName = productName;

this.category = category;

}

// getters

public int getId() {

return productId;

}

public String getName()

{

return productName;

}

public String getCategory()

{

return category;

}

*@Override*

public String toString()

{

return "\n" + productId + " - " + productName + " (" + category + ")\n";

}

}

**SearchFunctions.java**

package ECommerceSearchFunctions;

public class SearchFunctions

{

// Linear Search

public static Product linearSearch(Product products[],String name)

{

for(Product arr : products)

{

if(arr.getName().equalsIgnoreCase(name))

{

return arr;

}

}

return null;

}

// Binary Search

public static Product binarySearch(Product products[],String name)

{

int left = 0 , right = products.length - 1;

while(left <= right)

{

int mid = (left + right)/2;

int compare = products[mid].getName().compareToIgnoreCase(name);

if(compare == 0) return products[mid];

else if(compare < 0) left = mid + 1;

else right = mid - 1;

}

return null;

}

}

***Main.java***

package ECommerceSearchFunctions;

import java.util.Arrays;

public class Main

{

public static void main(String args[])

{

// Creating products directly in the array

Product[] products = {

new Product(101, "Laptop", "Electronics"),

new Product(102, "Shirt", "Fashion"),

new Product(103, "Coffee Maker", "Home"),

new Product(104, "Phone", "Electronics")

};

Product linear = SearchFunctions.*linearSearch*(products, "Phone");

System.***out***.println("\nLinear Search Result: " + (linear != null ? linear : "\nProduct not found"));

Arrays.*sort*(products,(a,b)-> a.getName().compareToIgnoreCase(b.getName()));

Product binary = SearchFunctions.*binarySearch*(products, "Coffee Maker");

System.***out***.println("Binary Search Result: " + (binary != null ? binary : "\nProduct not found"));

}

}

**OUTPUT:**

