// Custom Exception

class InvalidProductException extends Exception {

public InvalidProductException(String message) {

super(message);

}

}

// Product Class

class Product {

int productId;

String productName;

double price;

int quantity;

// Constructor with all fields

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

if (price < 0 || quantity < 0)

throw new InvalidProductException("Error: Price and Quantity must be non-negative.");

this.productId = productId;

this.productName = productName;

this.price = price;

this.quantity = quantity;

}

// Constructor with only productId and productName

public Product(int productId, String productName) {

this.productId = productId;

this.productName = productName;

this.price = 0.0;

this.quantity = 0;

}

// Overloaded method: full details

public void addProduct(int id, String name, double price, int quantity) throws InvalidProductException {

if (price < 0 || quantity < 0)

throw new InvalidProductException("Error: Price and Quantity must be non-negative.");

this.productId = id;

this.productName = name;

this.price = price;

this.quantity = quantity;

}

// Overloaded method: name and price

public void addProduct(String name, double price) throws InvalidProductException {

if (price < 0)

throw new InvalidProductException("Error: Price must be non-negative.");

this.productName = name;

this.price = price;

}

// Overloaded method: name, price, quantity

public void addProduct(String name, double price, int quantity) throws InvalidProductException {

if (price < 0 || quantity < 0)

throw new InvalidProductException("Error: Price and Quantity must be non-negative.");

this.productName = name;

this.price = price;

this.quantity = quantity;

}

public void displayProduct() {

System.out.println("ID: " + productId + ", Name: " + productName +

", Price: ₹" + price + ", Quantity: " + quantity);

}

}

// Main class

public class ProductManager {

public static void main(String[] args) {

try {

// Valid products

Product p1 = new Product(1, "Tablet", 25000, 3);

Product p2 = new Product(2, "Monitor");

Product p3 = new Product(3, "Printer", 7000, 2);

p2.addProduct("LED Monitor", 10499);

p3.addProduct("Laser Printer", 11500, 4);

p1.displayProduct();

p2.displayProduct();

p3.displayProduct();

// Invalid product to trigger exception

Product p4 = new Product(4, "Faulty Product", -1000, 5);

p4.displayProduct(); // Won’t execute if exception is thrown above

} catch (InvalidProductException e) {

System.out.println(e.getMessage());

}

}

}

