**1. InventoryManagementSystem**

**Main.java**

import java.util.HashMap;  
import java.util.Scanner;  
public class Main {  
  
 public static HashMap<Integer, Product> *productList*=new HashMap<>();  
 public static void main(String[] args) {  
 Scanner io=new Scanner(System.*in*);  
 int productId;  
 String productName;  
 int quantity;  
 double price;  
 while (true) {  
 System.*out*.println("Inventory Management System");  
 System.*out*.println("1. Add Product");  
 System.*out*.println("2. Update Product");  
 System.*out*.println("3. Delete Products");  
 System.*out*.print("Enter your choice: ");  
 int op=io.nextInt();  
 switch (op) {  
 case 1:  
 System.*out*.print("Product ID: ");  
 productId=io.nextInt();  
 System.*out*.print("Product Name: ");  
 productName=io.next();  
 System.*out*.print("Quantity: ");  
 quantity=io.nextInt();  
 System.*out*.print("Price: ");  
 price=io.nextDouble();  
 System.*out*.println(AddProduct.*add*(productId,productName,quantity,price));  
 break;  
 case 2:  
 System.*out*.print("Product ID: ");  
 productId=io.nextInt();  
 System.*out*.print("Quantity: ");  
 quantity=io.nextInt();  
 System.*out*.print("Price: ");  
 price=io.nextDouble();  
 System.*out*.println(UpdateProduct.*update*(productId, quantity, price));  
 break;  
 case 3:  
 System.*out*.print("Product ID: ");  
 productId=io.nextInt();  
 System.*out*.println(DeleteProduct.*delete*(productId));  
 break;  
 case 4:  
 return;  
 default:  
 System.*out*.println("Enter valid choice");  
 }  
 }  
 }  
}

**Product.java**

public class Product {  
 int productId;  
 String productName;  
 int quantity;  
 double price;  
 public Product(int productId, String productName, int quantity, double price) {  
 this.productId=productId;  
 this.productName=productName;  
 this.quantity=quantity;  
 this.price=price;  
 }  
}

**AddProduct.java**

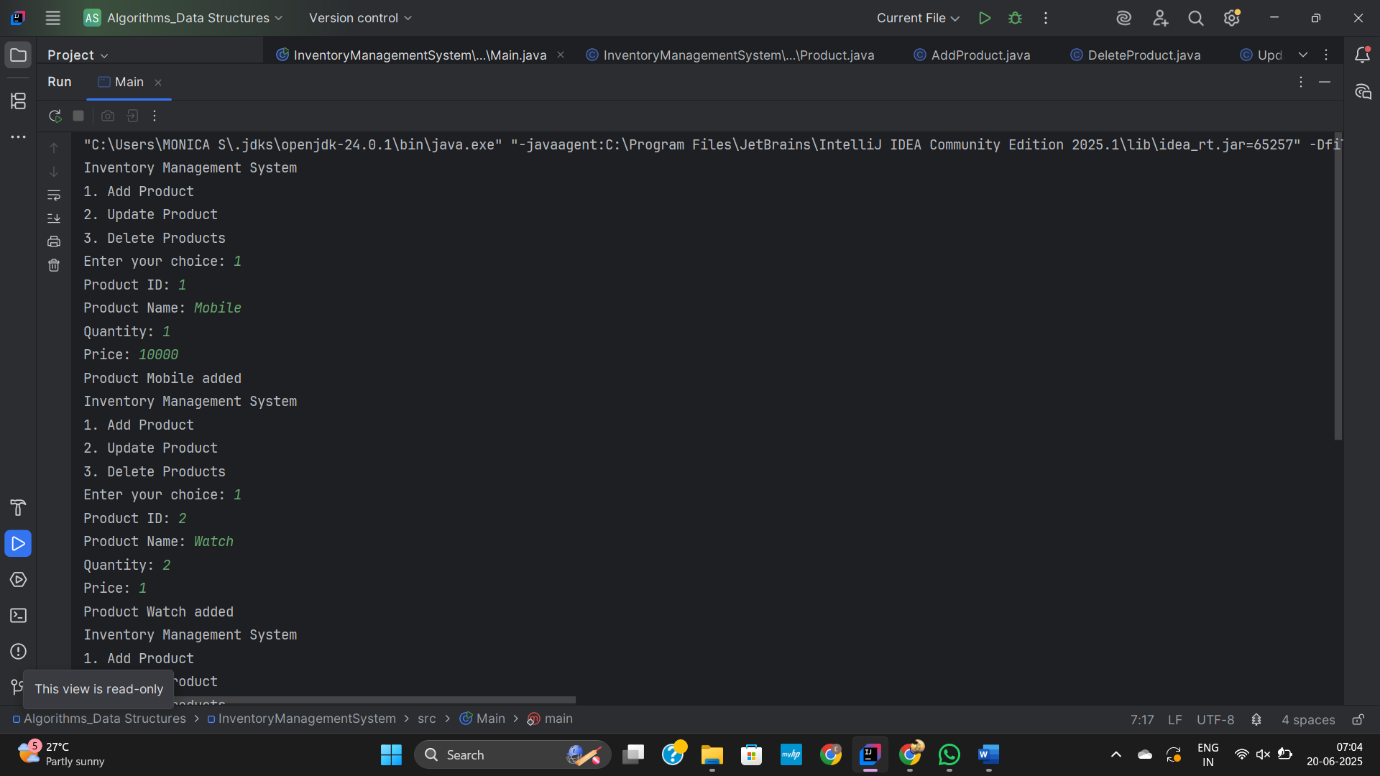
public class AddProduct {  
 public static String add(int productId, String productName, int quantity, double price) {  
 Product p=new Product(productId, productName, quantity ,price);  
 Main.*productList*.put(productId, p);  
 return "Product "+Main.*productList*.get(productId).productName+" added";  
 }  
}

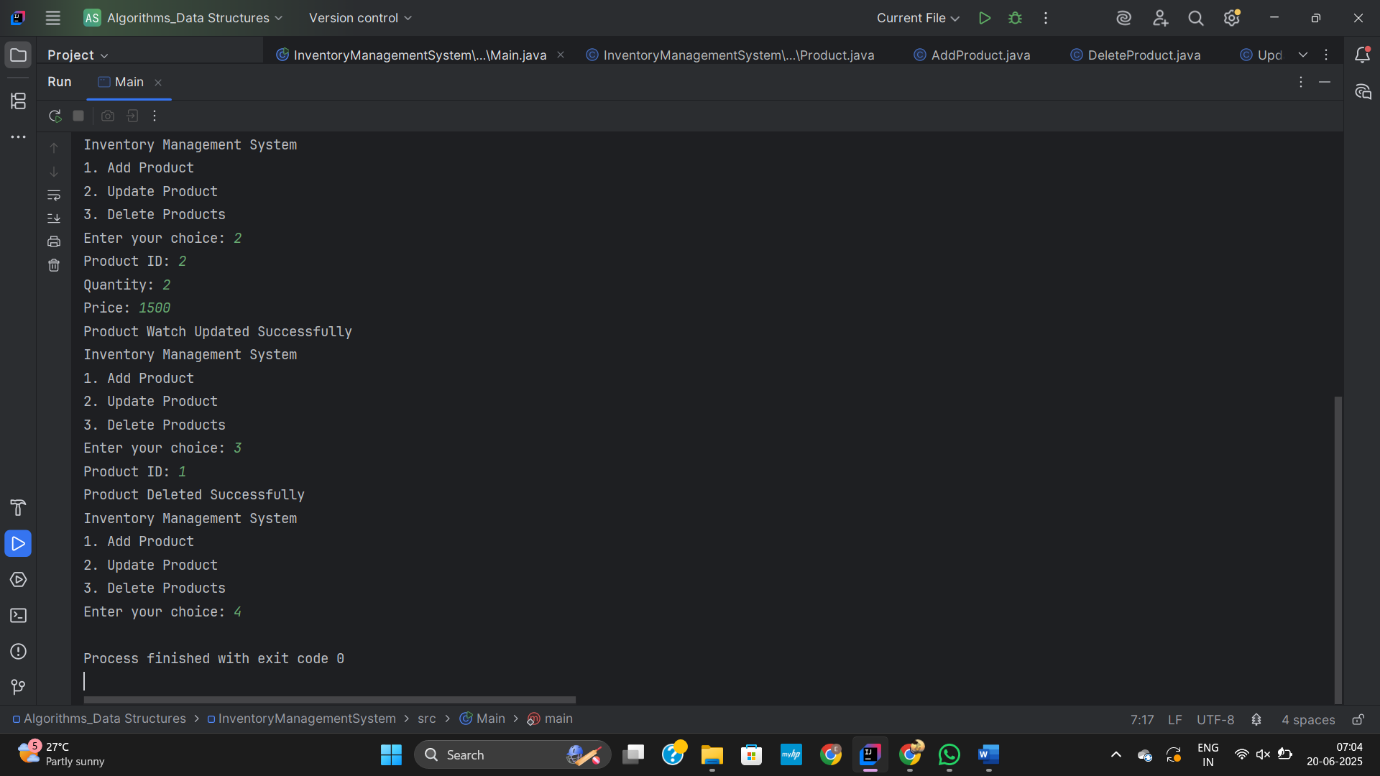
**DeleteProduct.java**

public class DeleteProduct {  
 public static String delete(int productId) {  
 if(Main.*productList*.containsKey(productId)) {  
 Main.*productList*.remove(productId);  
 }  
 return "Product Deleted Successfully";  
 }  
}

**UpdateProduct.java**

public class UpdateProduct {  
  
 public static String update(int productId, int quantity, double price) {  
 if(Main.*productList*.containsKey(productId)) {  
 Main.*productList*.get(productId).quantity=quantity;  
 Main.*productList*.get(productId).price=price;  
 }  
 else{  
 return "No Product found";  
 }  
 return "Product "+Main.*productList*.get(productId).productName+" Updated Successfully";  
 }  
}





**2. E-commercePlatformSearchFunction**

**Main.java**

import java.util.Arrays;  
import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Scanner io=new Scanner(System.*in*);  
 LinearSearch l=LinearSearch.*getInstance*();  
 BinarySearch b=BinarySearch.*getInstance*();  
 int productId;  
 String productName;  
 String category;  
 int quantity;  
 double amount;  
 System.*out*.print("Enter number of Elements: ");  
 int n=io.nextInt();  
 Product[] product=new Product[n];  
 for(int i=0;i<n;i++) {  
 System.*out*.print("Product ID: ");  
 productId= io.nextInt();  
 System.*out*.print("Product Name: ");  
 productName=io.next();  
 System.*out*.print("Category: ");  
 category=io.next();  
 System.*out*.print("Quantity: ");  
 quantity=io.nextInt();  
 System.*out*.print("Amount: ");  
 amount=io.nextDouble();  
 Product p=new Product(productId, productName, category, quantity, amount);  
 product[i]=p;  
 System.*out*.println("Added successfully");  
 }  
 while (true) {  
 System.*out*.println("Searching Operation");  
 System.*out*.println("1. Linear search\n2. Binary search\n3. Exit");  
 System.*out*.print("Enter your choice: ");  
 int ch=io.nextInt();  
 switch (ch) {  
 case 1:  
 System.*out*.print("Search product: ");  
 productName=io.next();  
 System.*out*.println(l.linearSearch(product, productName));  
 break;  
 case 2:  
 System.*out*.print("Search product: ");  
 productName=io.next();  
 System.*out*.println(b.binary(product, productName));  
 break;  
 case 3:  
 System.*out*.println("Bye...");  
 return;  
 default:  
 System.*out*.println("Enter valid choice");  
 }  
 }  
 }  
}

**Product.java**

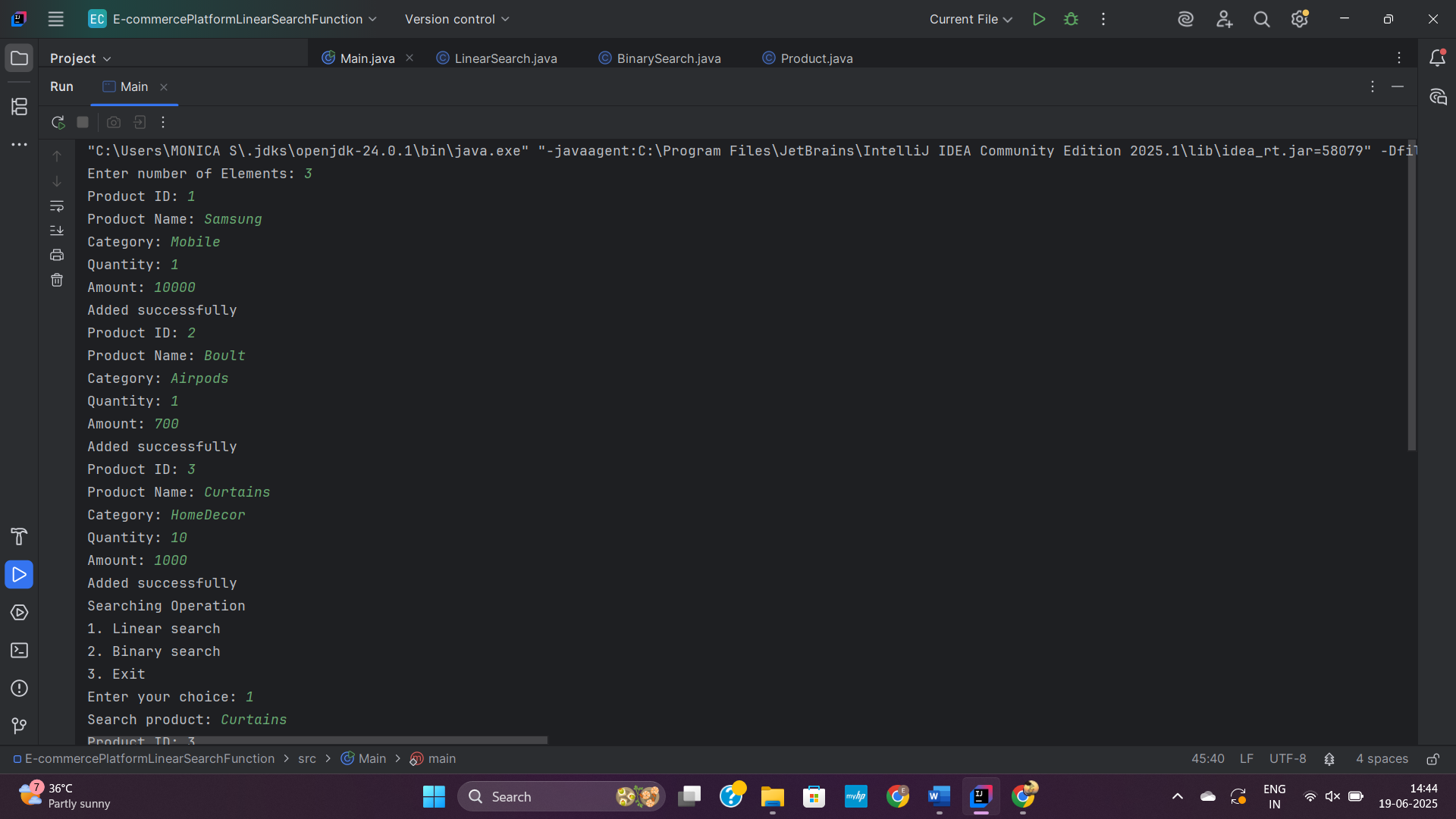
public class Product {  
 int productId;  
 String productName;  
 String category;  
 int quantity;  
 double amount;  
 public Product(int productId, String productName, String category, int quantity, double amount) {  
 this.productId=productId;  
 this.productName=productName;  
 this.category=category;  
 this.quantity=quantity;  
 this.amount=amount;  
 }  
}

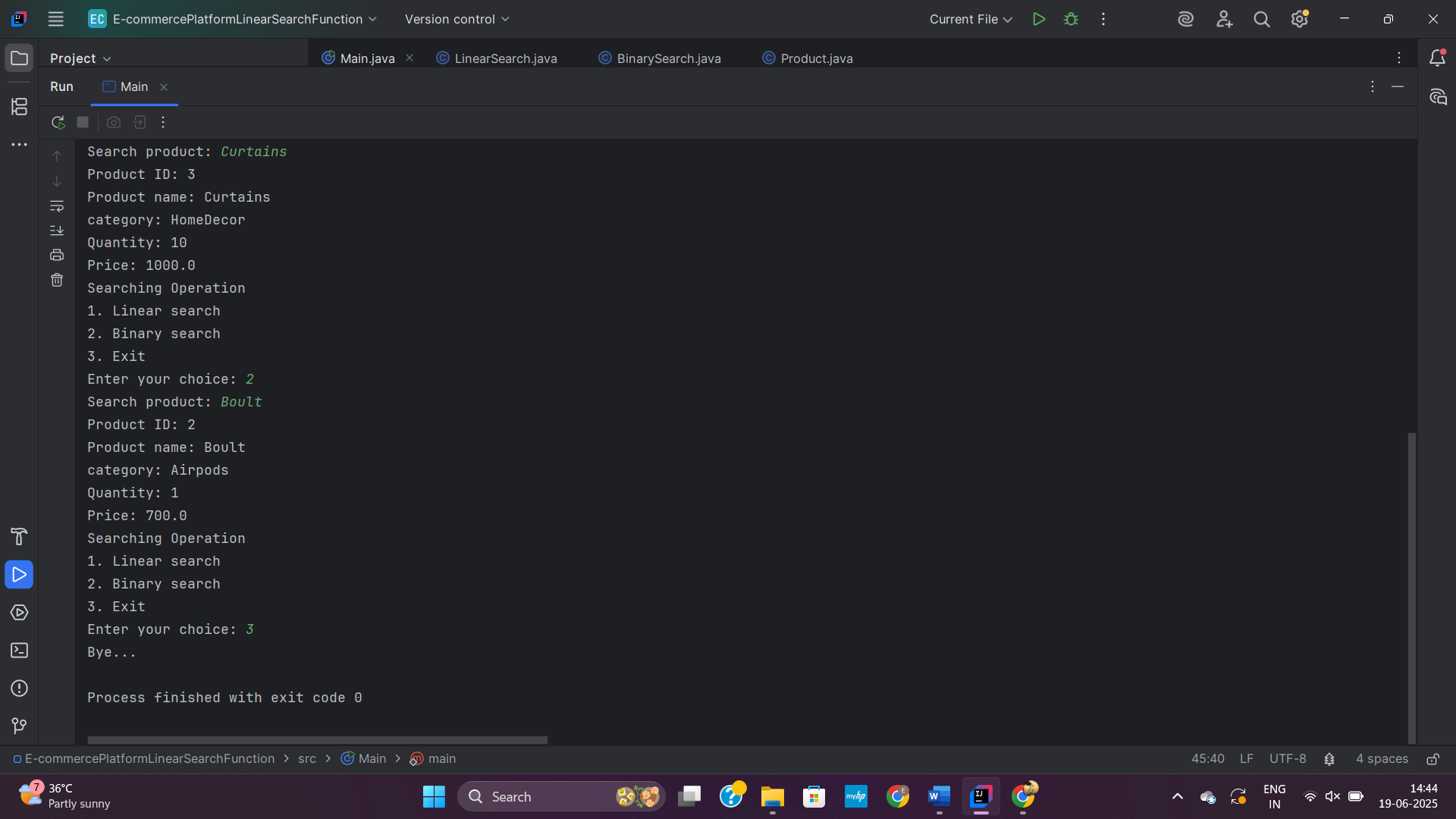
**BinarySearch.java**

import java.util.Arrays;  
import java.util.Comparator;  
public class BinarySearch {  
 private BinarySearch() {  
 }  
 public static BinarySearch getInstance() {  
 return new BinarySearch();  
 }  
 public String binary(Product[] products, String name) {  
 Arrays.*sort*(products, new Comparator<Product>() {  
 public int compare(Product p1, Product p2) {  
 return p1.productName.compareToIgnoreCase(p2.productName);  
 }  
 });  
 int low = 0, high = products.length - 1;  
 while (low <= high) {  
 int mid = (low + high) / 2;  
 int compare = name.compareToIgnoreCase(products[mid].productName);  
 if (compare == 0)  
 return "Product ID: "+products[mid].productId+"\nProduct name: "+products[mid].productName+  
 "\ncategory: "+products[mid].category+"\nQuantity: "+products[mid].quantity+"\nPrice: "+products[mid].amount;  
 else if (compare < 0)  
 high = mid - 1;  
 else  
 low = mid + 1;  
 }  
 return null;  
 }  
}

**LinearSearch.java**

public class LinearSearch {  
 private LinearSearch() {  
 }  
 public static LinearSearch getInstance() {  
 return new LinearSearch();  
 }  
 public String linearSearch(Product[] products, String name) {  
 for (int i = 0; i < products.length; i++) {  
 if (products[i].productName.equalsIgnoreCase(name)) {  
 return "Product ID: "+products[i].productId+"\nProduct name: "+products[i].productName+  
 "\ncategory: "+products[i].category+"\nQuantity: "+products[i].quantity+"\nPrice: "+products[i].amount;  
 }  
 }  
 return null;  
 }  
}





**3. SortingCustomerOrder**

**Main.java**

import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
  
 Scanner io=new Scanner(System.in);  
 int OrderId;  
 String OrderName;  
 double totalPrice;  
 System.out.print("Enter number of Orders: ");  
 int n=io.nextInt();  
 Order[] order=new Order[n];  
 for(int i=0;i<n;i++) {  
 System.out.print("Order ID: ");  
 OrderId=io.nextInt();  
 System.out.print("Order name: ");  
 OrderName=io.next();  
 System.out.print("Total price: ");  
 totalPrice=io.nextDouble();  
 Order o=new Order(OrderId, OrderName, totalPrice);  
 order[i]=o;  
 }  
 while (true) {  
 System.out.println("Sorting Operation");  
 System.out.println("1. Bubble sort\n2. Quick sort\n3. Exit");  
 System.out.print("Enter your choice(1/2/3): ");  
 int ch=io.nextInt();  
 switch (ch) {  
 case 1:  
 BubbleSort.sort(order);  
 break;  
 case 2:  
 QuickSort.sort(order, 0, n - 1);  
 System.out.println("Sorted using Quick Sort:");  
 QuickSort.printOrders(order);  
 break;

case 3:  
 System.out.println("Bye...");  
 return;  
 default:  
 }  
 }  
 }  
}

**Order.java**

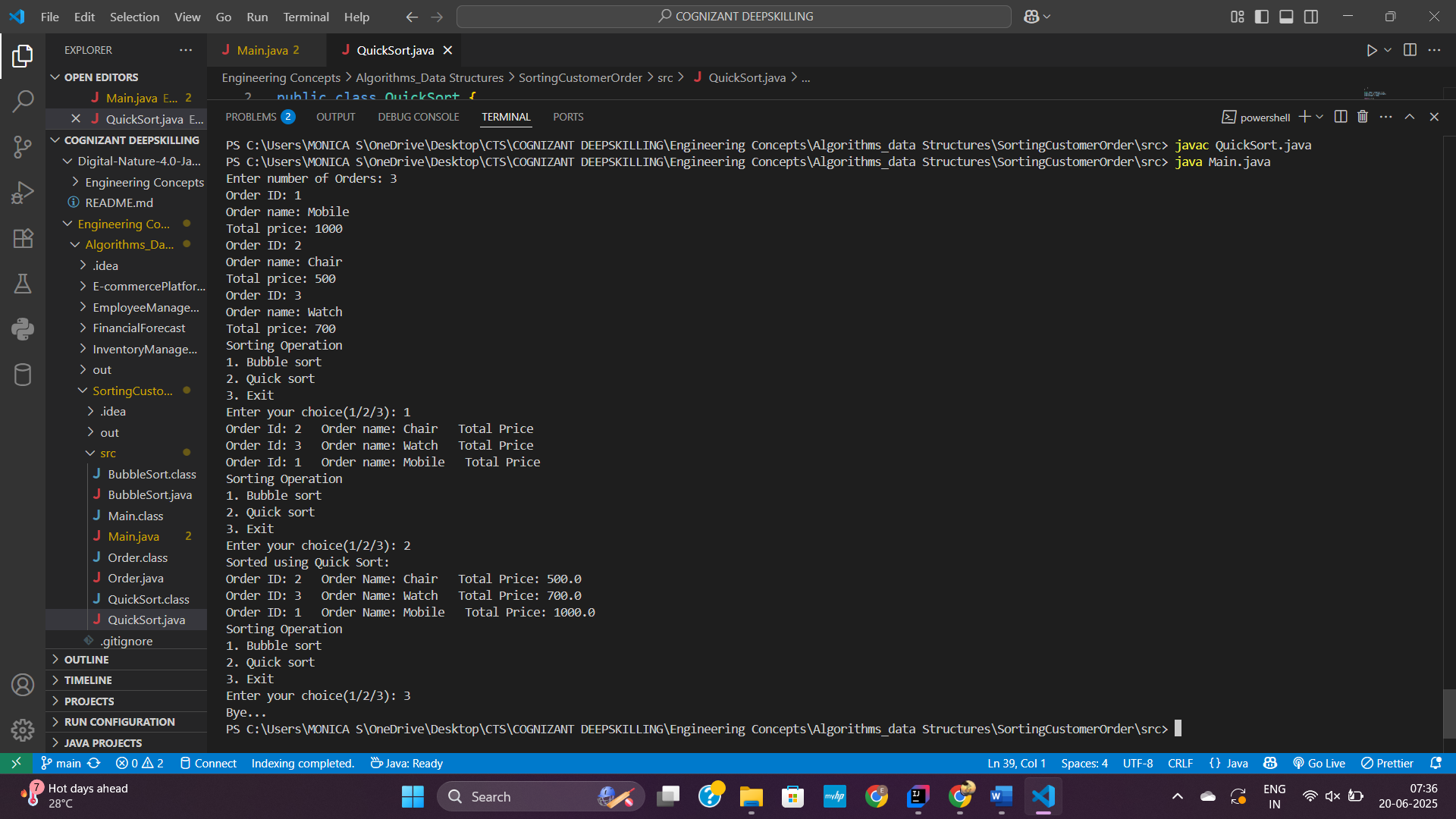
public class Order {  
  
 int OrderId;  
 String OrderName;  
 double totalPrice;  
  
 public Order(int OrderId, String OrderName, double totalPrice){  
 this.OrderId=OrderId;  
 this.OrderName=OrderName;  
 this.totalPrice=totalPrice;  
 }  
  
}

**BubbleSort.java**

public class BubbleSort {  
 public static void sort(Order[] orders) {  
 int n=orders.length;  
 for (int i=0;i<n-1;i++) {  
 for (int j=0;j<n-i-1;j++) {  
 if(orders[j].totalPrice>orders[j+1].totalPrice){  
 Order temp=orders[j];  
 orders[j]=orders[j+1];  
 orders[j+1]=temp;  
 }  
 }  
 }  
 for(int i=0;i<n;i++) {  
 System.out.print("Order Id: "+orders[i].OrderId+" Order name: "+orders[i].OrderName+" Total Price\n");  
 }  
 }  
}

**QuickSort.java**

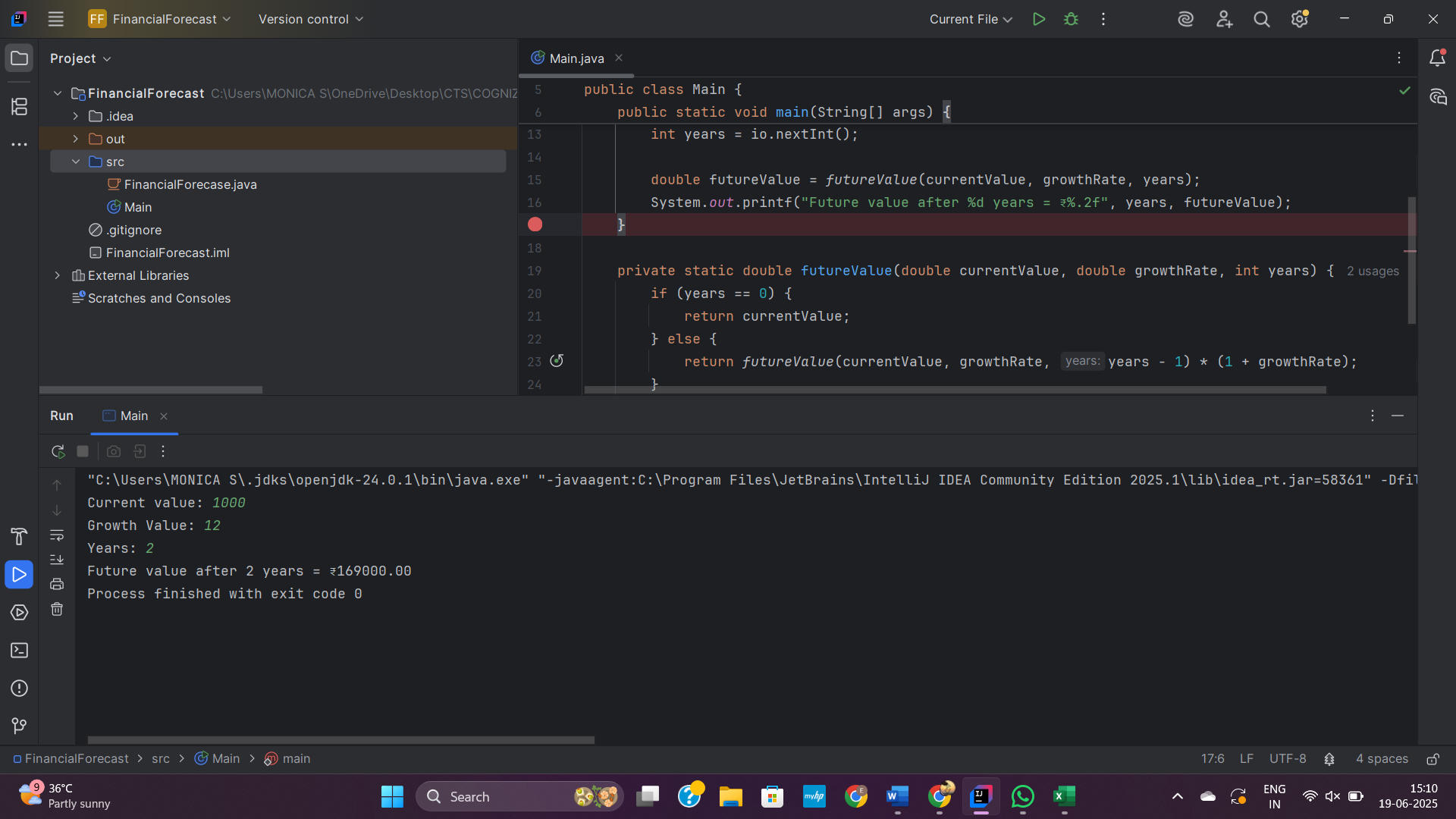
public class QuickSort {  
  
 public static void sort(Order[] orders, int low, int high) {  
 if (low < high) {  
 int pivotIndex = partition(orders, low, high);  
 sort(orders, low, pivotIndex - 1);  
 sort(orders, pivotIndex + 1, high);  
 }  
  
 }  
  
 private static int partition(Order[] orders, int low, int high) {  
 double pivot = orders[high].totalPrice;  
 int i = low - 1;  
  
 for (int j = low; j < high; j++) {  
 if (orders[j].totalPrice < pivot) {  
 i++;  
 Order temp = orders[i];  
 orders[i] = orders[j];  
 orders[j] = temp;  
 }  
 }  
  
 Order temp = orders[i + 1];  
 orders[i + 1] = orders[high];  
 orders[high] = temp;  
  
 return i + 1;  
 }  
  
 public static void printOrders(Order[] orders) {  
 for (Order o : orders) {  
 System.out.println("Order ID: " + o.OrderId + " Order Name: " + o.OrderName + " Total Price: " + o.totalPrice);  
 }  
 }  
}

****

**4. FinancialForecast**

**Main.java**

import java.util.Scanner;  
  
//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or  
// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.  
public class Main {  
 public static void main(String[] args) {  
 Scanner io=new Scanner(System.in);  
 System.out.print("Current value: ");  
 double currentValue = io.nextDouble();  
 System.out.print("Growth Value: ");  
 double growthRate = io.nextDouble();  
 System.out.print("Years: ");  
 int years = io.nextInt();  
  
 double futureValue = futureValue(currentValue, growthRate, years);  
 System.out.printf("Future value after %d years = ₹%.2f", years, futureValue);  
 }  
  
 private static double futureValue(double currentValue, double growthRate, int years) {  
 if (years == 0) {  
 return currentValue;  
 } else {  
 return futureValue(currentValue, growthRate, years - 1) \* (1 + growthRate);  
 }  
 }  
}

****