**Exercise 1: Inventory Management System**

**package ex\_1;**

**import java.util.HashMap;**

**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;**

**}**

**public void display() {**

**System.*out*.println("Product ID : " + productId);**

**System.*out*.println("Product Name : " + productName);**

**System.*out*.println("Quantity : " + quantity);**

**System.*out*.println("Price : ₹" + price);**

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

**}**

**}**

**public class Demo {**

**static HashMap<Integer, Product> *inventory* = new HashMap<>();**

**public static void addProduct(Product product) {**

**if (*inventory*.containsKey(product.productId)) {**

**System.*out*.println("Product ID already exists!");**

**} else {**

***inventory*.put(product.productId, product);**

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

**}**

**}**

**public static void updateProduct(int productId, int quantity, double price) {**

**if (*inventory*.containsKey(productId)) {**

**Product p = *inventory*.get(productId);**

**p.quantity = quantity;**

**p.price = price;**

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

**} else {**

**System.*out*.println("Product not found.");**

**}**

**}**

**public static void deleteProduct(int productId) {**

**if (*inventory*.containsKey(productId)) {**

***inventory*.remove(productId);**

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

**} else {**

**System.*out*.println("Product not found.");**

**}**

**}**

**public static void displayInventory() {**

**if (*inventory*.isEmpty()) {**

**System.*out*.println("Inventory is empty.");**

**} else {**

**System.*out*.println("Inventory List:");**

**for (Product p : *inventory*.values()) {**

**p.display();**

**}**

**}**

**}**

**public static void main(String[] args) {**

**Product p1 = new Product(1, "Notebook", 100, 25.5);**

**Product p2 = new Product(2, "Pen", 200, 10.0);**

**Product p3 = new Product(3, "Pencil", 300, 5.0);**

***addProduct*(p1);**

***addProduct*(p2);**

***addProduct*(p3);**

**System.*out*.println("\nInitial Inventory:");**

***displayInventory*();**

***updateProduct*(2, 250, 9.5);**

**System.*out*.println("\nAfter Updating Product ID 2:");**

***displayInventory*();**

***deleteProduct*(1);**

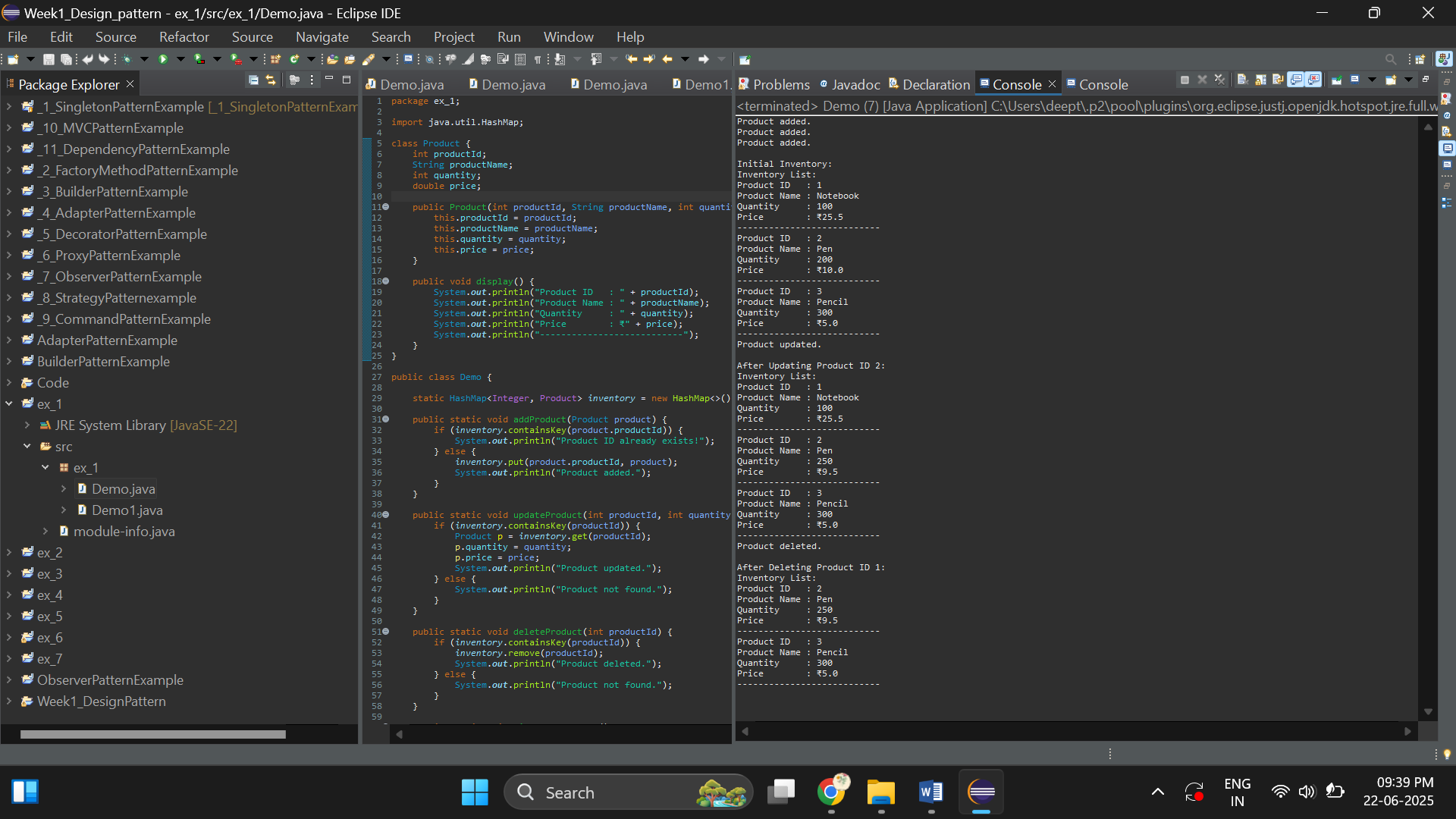
**System.*out*.println("\nAfter Deleting Product ID 1:");**

***displayInventory*();**

**}**

**}**

**Output:**

****

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

**package ex\_2;**

**import java.util.\*;**

**class Product{**

**protected int productId;**

**protected String productName;**

**protected String category;**

**Product(int productId,String productName,String category){**

**this.productId=productId;**

**this.productName=productName;**

**this.category=category;**

**}**

**public int getProductId() {**

**return productId;**

**}**

**public String getProductName() {**

**return productName;**

**}**

**public String getCategory() {**

**return category;**

**}**

**public void display() {**

**System.*out*.println("Product Details:");**

**System.*out*.println("Product Id :"+getProductId());**

**System.*out*.println("Product Name :"+getProductName());**

**System.*out*.println("Product Category :"+getCategory());**

**}**

**}**

**//Search productName by productId->UsingLinearSearch**

**class SearchProduct{**

**List<Product> list=new ArrayList<>();**

**public void addAllProducts(Product p) {**

**list.add(p);**

**}**

**public void find\_method1\_linearly(int target) {**

**boolean found=false;**

**for(Product p:list) {**

**if(target==p.getProductId()) {**

**System.*out*.println("Search element is found!!");**

**found=true;**

**p.display();**

**break;**

**}**

**}**

**if(!found) System.*out*.println("Not found");**

**}**

**public void Sort() {**

**int n = list.size();**

**//outerloop**

**for(int i = 0; i < n - 1; i++) {**

**//innerloop**

**for(int j = 0; j < n - i - 1; j++) {**

**if(list.get(j).getProductId() > list.get(j + 1).getProductId()) {**

**Product temp = list.get(j);**

**list.set(j, list.get(j + 1));**

**list.set(j + 1, temp);**

**}**

**}**

**}**

**System.*out*.println("After sorting:");**

**for(Product p:list) {**

**p.display();**

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

**}**

**}**

**public void find\_method1\_Binary(int target) {**

**Sort();**

**int n=list.size();**

**int left=0;**

**int right=n-1;**

**while(left<=right) {**

**int mid=(left+right)/2;**

**int midId=list.get(mid).getProductId();**

**if(midId==target) {**

**System.*out*.println("Search product is found (Binary Search)");**

**list.get(mid).display();**

**break;**

**}else if(target > midId) {**

**left=mid+1;**

**}else {**

**right=mid-1;**

**}**

**}**

**System.*out*.println("Search element is not found!");**

**}**

**}**

**public class Demo {**

**public static void main(String[] args) {**

**Product p1=new Product(101,"Mobile","Electronics");**

**Product p2=new Product(102,"Bluetooth","Electronics");**

**Product p3=new Product(201,"Pepsi","Snacks");**

**SearchProduct store=new SearchProduct();**

**store.addAllProducts(p1);**

**store.addAllProducts(p2);**

**store.addAllProducts(p3);**

**store.find\_method1\_linearly(101);//Search element is found!!**

**store.find\_method1\_linearly(78);//not found**

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

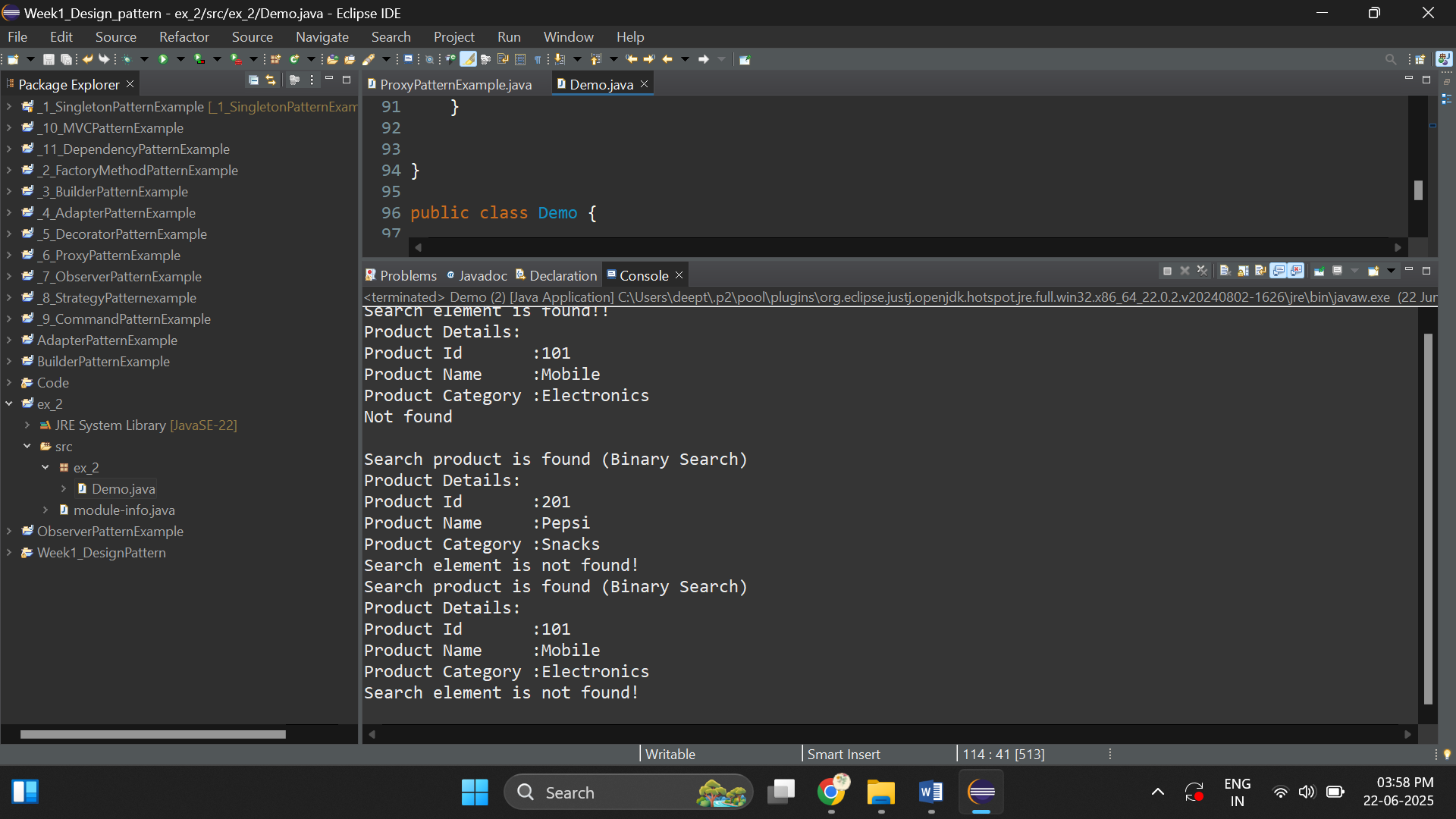
**store.find\_method1\_Binary(201);**

**store.find\_method1\_Binary(101);**

**}**

**}**

**Output:**

****

**Exercise 3: Sorting Customer Orders**

**package ex\_3;**

**import java.util.\*;**

**class Order{**

**//orderId, customerName, and totalPrice.**

**protected int orderId;**

**protected String customerName;**

**protected double totalPrice;**

**public Order(int orderId,String customerName,double totalPrice) {**

**this.orderId=orderId;**

**this.customerName=customerName;**

**this.totalPrice=totalPrice;**

**}**

**public int getOrderId() {**

**return orderId;**

**}**

**public String getCustomerName() {**

**return customerName;**

**}**

**public double getToatlPrice() {**

**return totalPrice;**

**}**

**public void display() {**

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

**System.*out*.println("Order Id :"+getOrderId());**

**System.*out*.println("Customer Name :"+getCustomerName());**

**System.*out*.println("Total price :"+getToatlPrice());**

**}**

**}**

**class sortByOrderId{**

**List<Order> orders=new ArrayList<>();**

**public void add(Order o) {**

**orders.add(o);**

**}**

**//method-1(bubble sort)**

**public void bubble\_sort() {**

**int n=orders.size();**

**//outerloop**

**for(int i=0;i<n-1;i++) {**

**//inner loop**

**for(int j=0;j<n-i-1;j++) {**

**if(orders.get(j).getOrderId() > orders.get(j+1).getOrderId()) {**

**//swap**

**Order temp=orders.get(j);**

**orders.set(j,orders.get(j+1));**

**orders.set(j+1,temp);**

**}**

**}**

**}**

**for(Order order:orders) {**

**order.display();**

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

**}**

**}**

**public void quick\_sort() {**

**quick\_sort(0,orders.size()-1);**

**for(Order order:orders) {**

**order.display();**

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

**}**

**}**

**private void quick\_sort(int low,int high) {**

**//base condition**

**if(low>=high) {**

**return;**

**}**

**int start=low;**

**int end=high;**

**while(start<=end) {**

**int pivot=(start+(end-start))/2;**

**while(orders.get(start).getOrderId() < orders.get(pivot).getOrderId()) {**

**start++;**

**}**

**while(orders.get(end).getOrderId()>orders.get(pivot).getOrderId() ) {**

**end--;**

**}**

**//if violation occurs**

**if(start<=end) {**

**Order temp=orders.get(start);**

**orders.set(start,orders.get(end));**

**orders.set(end, temp);**

**start++;**

**end--;**

**}**

**}**

**//recursive call**

**quick\_sort(low, end);**

**quick\_sort(start, high);**

**}**

**}**

**public class Demo {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Order o1=new Order(5,"Deepthikha",25.0);**

**Order o2=new Order(6,"Jothi",34.0);**

**Order o3=new Order(3,"Rama kumar",76.8);**

**sortByOrderId store=new sortByOrderId();**

**store.add(o1);**

**store.add(o2);**

**store.add(o3);**

**System.*out*.println("Bubble sort:");**

**//sort by order id**

**store.bubble\_sort();**

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

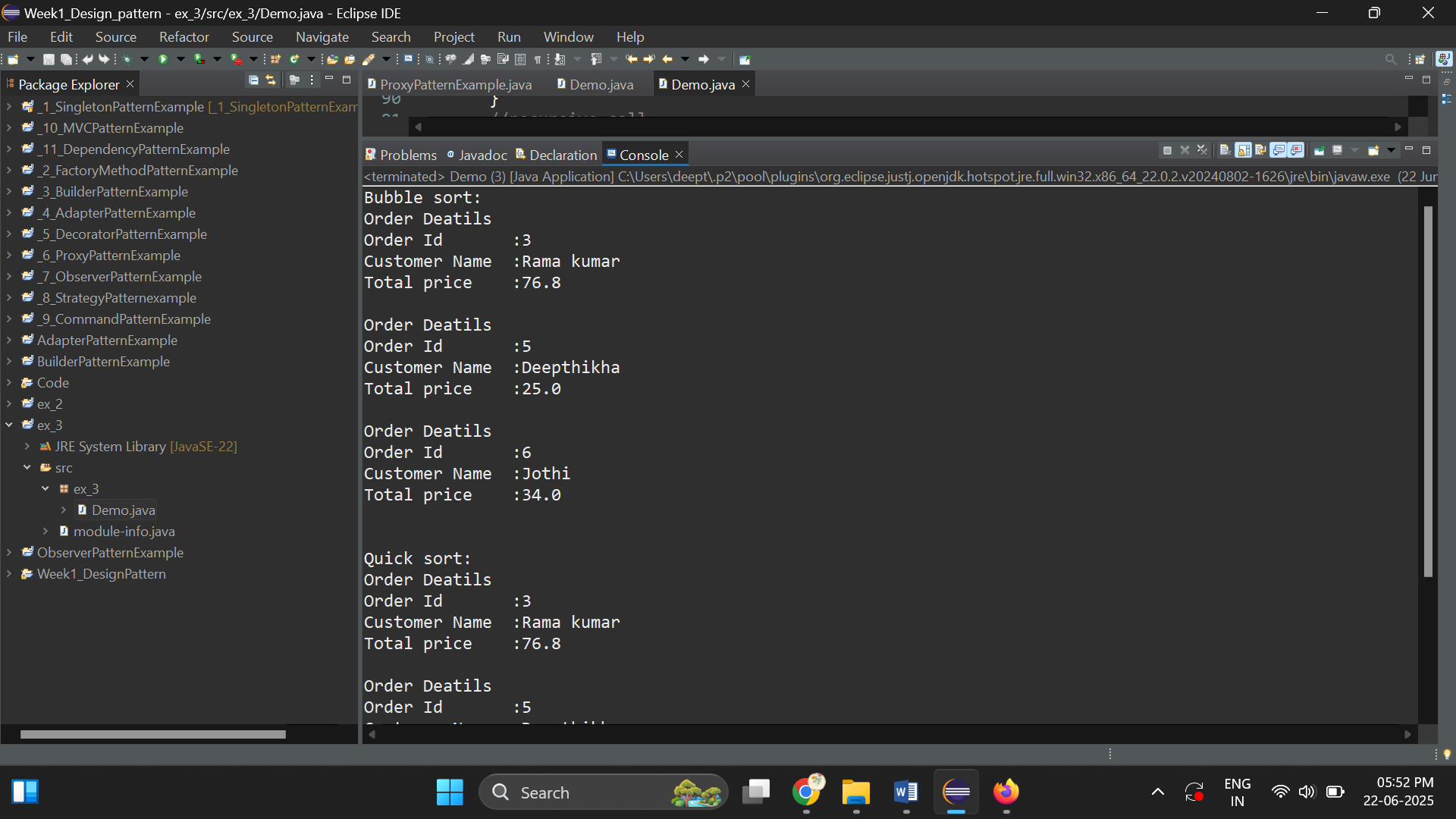
**System.*out*.println("Quick sort:");**

**store.quick\_sort();**

**}**

**}**

**Output:**

****

**Exercise 4: Employee Management System**

**package ex\_4;**

**import java.util.\*;**

**class Employee {**

**protected int employeeId;**

**protected String name;**

**protected String position;**

**protected double salary;**

**Employee(int employeeId, String name, String position, double salary) {**

**this.employeeId = employeeId;**

**this.name = name;**

**this.position = position;**

**this.salary = salary;**

**}**

**public int getEmployeeId() { return employeeId; }**

**public void setEmployeeId(int employeeId) { this.employeeId = employeeId; }**

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

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

**public String getPosition() { return position; }**

**public void setPosition(String position) { this.position = position; }**

**public double getSalary() { return salary; }**

**public void setSalary(double salary) { this.salary = salary; }**

**public void display() {**

**System.*out*.println("Employee Details:");**

**System.*out*.println("Employee ID : " + getEmployeeId());**

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

**System.*out*.println("Position : " + getPosition());**

**System.*out*.println("Salary : " + getSalary());**

**}**

**}**

**class Employee\_records {**

**List<Employee> records = new ArrayList<>();**

**public void add(Employee e) {**

**records.add(e);**

**}**

**public void delete(Employee e,String position) {**

**if (position.equalsIgnoreCase("admin")) {**

**records.remove(e);**

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

**} else {**

**System.*out*.println("Access Denied: Only Admin can delete records.");**

**}**

**}**

**public void search(int employeeId) {**

**boolean found = false;**

**for (Employee employee : records) {**

**if (employee.getEmployeeId() == employeeId) {**

**System.*out*.println("Employee Found!!");**

**employee.display();**

**found = true;**

**break;**

**}**

**}**

**if (!found) {**

**System.*out*.println("Employee Not Found!!");**

**}**

**}**

**public void traverse() {**

**System.*out*.println("All Employee Records:");**

**for (Employee e : records) {**

**e.display();**

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

**}**

**}**

**}**

**public class Demo {**

**public static void main(String[] args) {**

**Employee e1 = new Employee(1, "Ravi", "admin", 50000.0);**

**Employee e2 = new Employee(2, "Ramakumar", "admin", 50000.0);**

**Employee e3 = new Employee(3, "Jothi", "guest", 300000.0);**

**Employee e4 = new Employee(4, "Deepthikha", "guest", 300000.0);**

**Employee\_records record = new Employee\_records();**

**record.add(e1);**

**record.add(e2);**

**record.add(e3);**

**record.add(e4);**

**//Traverse**

**record.traverse();**

**System.*out*.println("Search Result:");**

**record.search(3);**

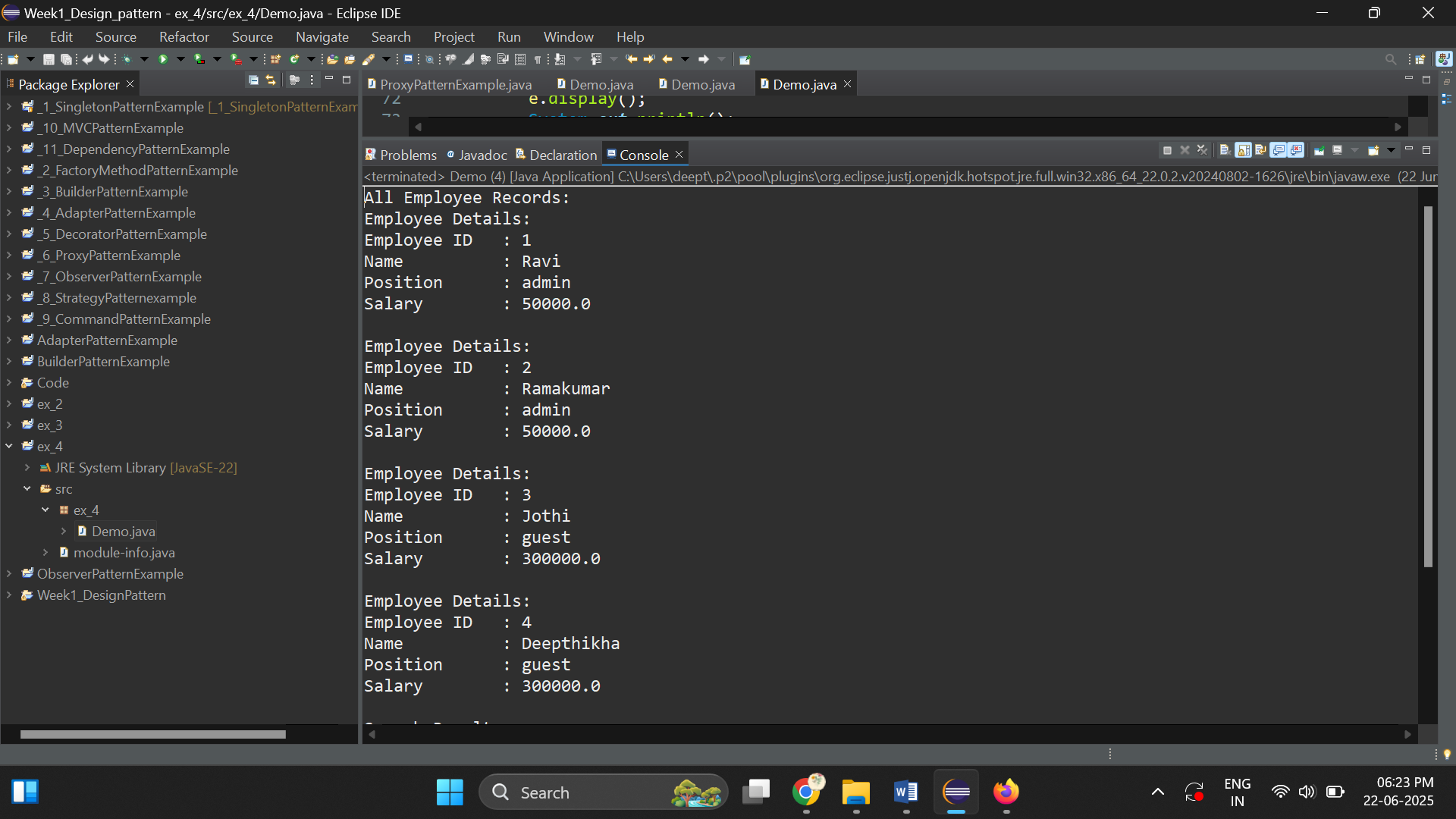
**System.*out*.println();**

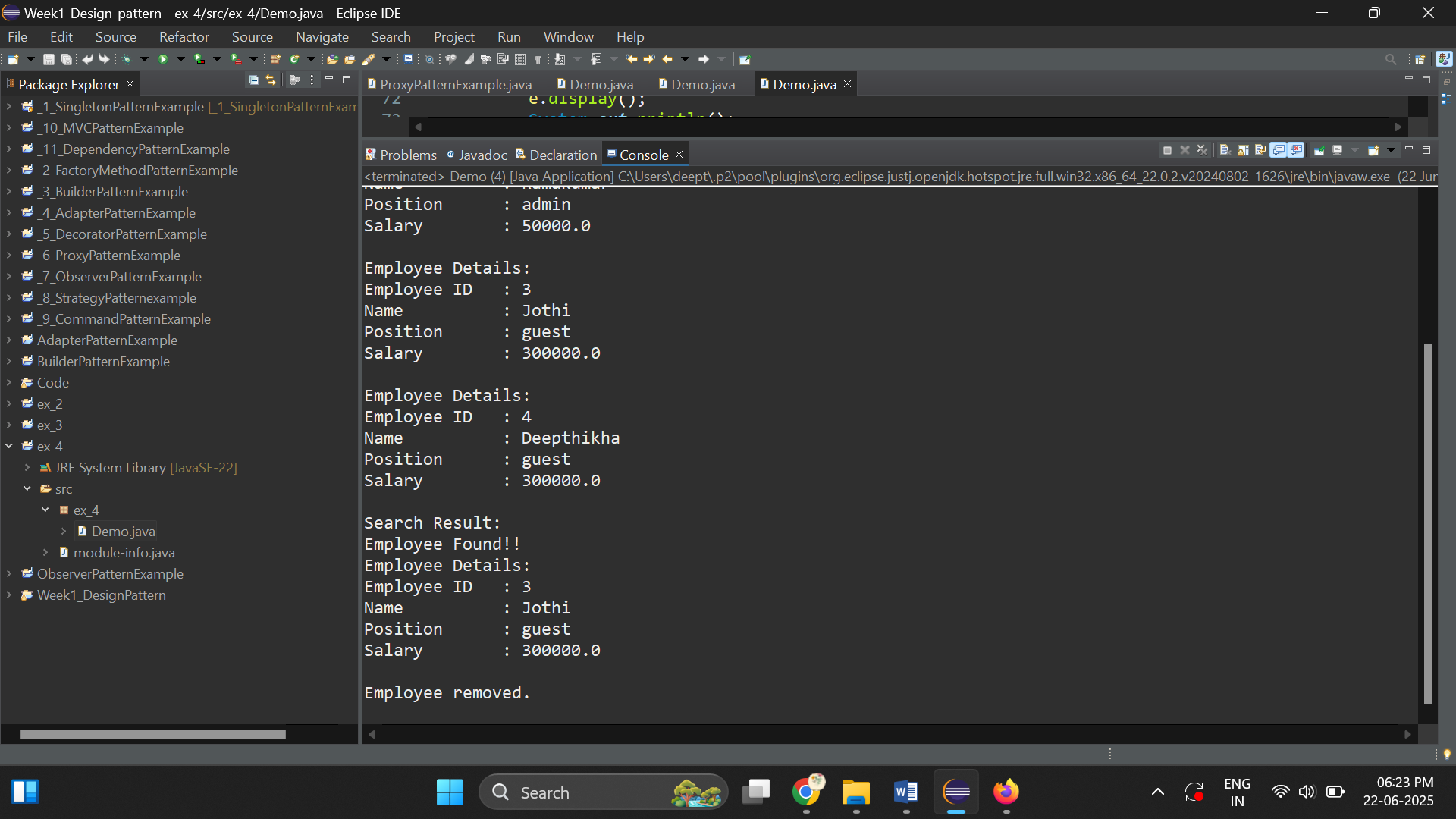
**record.delete(e4,"admin");**

**}**

**}**

**Output:**

****

****

**Exercise 5: Task Management System**

**package ex\_5;**

**class Task {**

**public int taskId;**

**public String taskName;**

**public String status;**

**public Task(int taskId, String taskName, String status) {**

**this.taskId = taskId;**

**this.taskName = taskName;**

**this.status = status;**

**}**

**public int getTaskId() { return taskId; }**

**public String getTaskName() { return taskName; }**

**public String getStatus() { return status; }**

**public void display() {**

**System.*out*.println("Task Details:");**

**System.*out*.println("Task Id : " + getTaskId());**

**System.*out*.println("Task Name : " + getTaskName());**

**System.*out*.println("Status : " + getStatus());**

**}**

**}**

**class Node {**

**Task task;**

**Node next;**

**public Node(Task task) {**

**this.task = task;**

**this.next = null;**

**}**

**}**

**public class Demo {**

**public static Node *head*;**

**public static void addAtEnd(Task task) {**

**Node newNode = new Node(task);**

**if (*head* == null) {**

***head* = newNode;**

**} else {**

**Node ptr = *head*;**

**while (ptr.next != null) {**

**ptr = ptr.next;**

**}**

**ptr.next = newNode;**

**}**

**}**

**public static Node delete(int taskId) {**

**if (*head* == null) {**

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

**return *head*;**

**}**

**if (*head*.task.getTaskId() == taskId) {**

***head* = *head*.next;**

**System.*out*.println("Deleted task with ID: " + taskId);**

**return *head*;**

**}**

**Node ptr = *head*;**

**while (ptr.next != null && ptr.next.task.getTaskId() != taskId) {**

**ptr = ptr.next;**

**}**

**if (ptr.next == null) {**

**System.*out*.println("Task ID " + taskId + " not found.");**

**} else {**

**ptr.next = ptr.next.next;**

**System.*out*.println("Deleted task with ID: " + taskId);**

**}**

**return *head*;**

**}**

**public static void displayAllTasks() {**

**if (*head* == null) {**

**System.*out*.println("No tasks to display.");**

**return;**

**}**

**Node ptr = *head*;**

**while (ptr != null) {**

**ptr.task.display();**

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

**ptr = ptr.next;**

**}**

**}**

**public static void searchById(int taskId) {**

**if (*head* == null) {**

**System.*out*.println("No tasks to search.");**

**return;**

**}**

**Node ptr = *head*;**

**boolean found = false;**

**while (ptr != null) {**

**if (ptr.task.getTaskId() == taskId) {**

**System.*out*.println("Task found with ID " + taskId + ":");**

**ptr.task.display();**

**found = true;**

**break;**

**}**

**ptr = ptr.next;**

**}**

**if (!found) {**

**System.*out*.println("Task with ID " + taskId + " not found.");**

**}**

**}**

**public static void searchByTaskId(int taskId) {**

**if (*head* == null) {**

**System.*out*.println("No tasks to search.");**

**return;**

**}**

**Node ptr = *head*;**

**boolean found = false;**

**//linear search**

**while (ptr != null) {**

**if (ptr.task.getTaskId()==taskId) {**

**System.*out*.println("Task found "+taskId);**

**ptr.task.display();**

**found = true;**

**break;**

**}**

**ptr = ptr.next;**

**}**

**if (!found) {**

**System.*out*.println("Not found.");**

**}**

**}**

**public static void main(String[] args) {**

**Task task1 = new Task(1, "Drink Water", "Completed");**

**Task task2 = new Task(2, "Go Walk", "Pending");**

**Task task3 = new Task(3, "Read Book", "In Progress");**

***addAtEnd*(task1);**

***addAtEnd*(task2);**

***addAtEnd*(task3);**

**System.*out*.println("All tasks:");**

***displayAllTasks*();**

**System.*out*.println("Deleting task with ID 2...");**

***delete*(2);**

**System.*out*.println("All tasks after deletion:");**

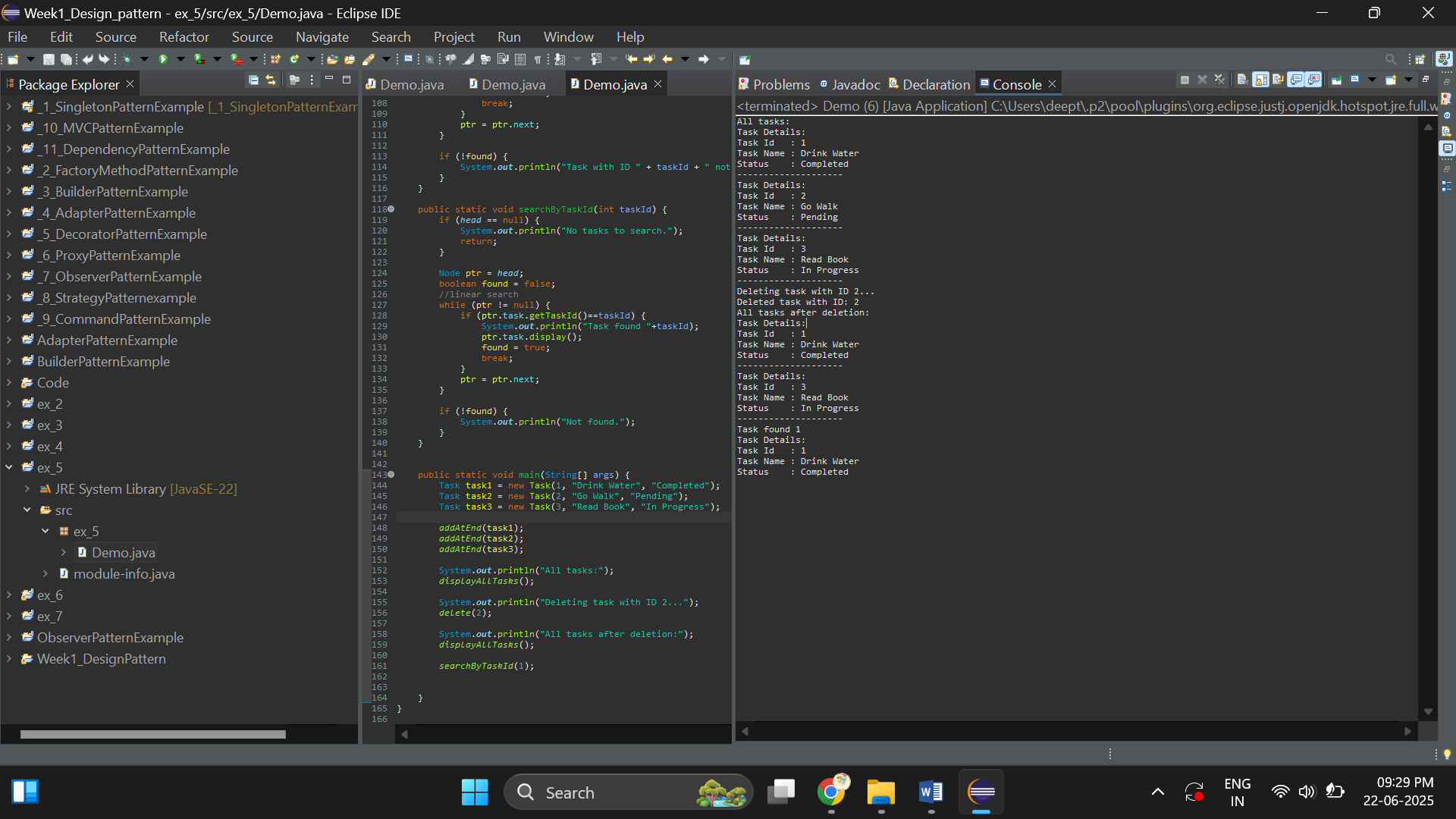
***displayAllTasks*();**

***searchByTaskId*(1);**

**}**

**}**

**Output:**

****

**Exercise 6: Library Management System**

**package ex\_6;**

**import java.util.\*;**

**class Book{**

**public int bookId;**

**public String title;**

**public String author;**

**public Book(int bookId,String title,String author) {**

**this.bookId=bookId;**

**this.title=title;**

**this.author=author;**

**}**

**public int getBookId() {return bookId;}**

**public String getBookTitle() {return title;}**

**public String getBookAuthor() {return author;}**

**public void display() {**

**System.*out*.println("Book details :");**

**System.*out*.println("Book Id :"+getBookId());**

**System.*out*.println("Book Title :"+getBookTitle());**

**System.*out*.println("Book Author:"+getBookAuthor());**

**}**

**}**

**class Search{**

**// Book book;**

**List<Book> books=new ArrayList<>();**

**public void add(Book book) {**

**books.add(book);**

**}**

**//method-1**

**public void linearSearch(String title) {**

**boolean found=false;**

**for(Book book:books) {**

**if(book.getBookTitle().equalsIgnoreCase(title)) {**

**System.*out*.println("Found!! -(linear Search)"+title);**

**book.display();**

**found=true;**

**break;**

**}**

**}**

**if(!found) {System.*out*.println("Not found!!");}**

**}**

**public void BinarySearch(String title) {**

**int target=0;**

**for(Book book:books) {**

**String current=book.getBookTitle();**

**if(current.equalsIgnoreCase(title)) {**

**target=book.getBookId();**

**}**

**}**

**BinarySearch(title,target);**

**}**

**private void BinarySearch(String title,int target) {**

**int size=books.size();**

**int left=0;**

**int right=size-1;**

**while(left<=right) {**

**int mid=(left+right)/2;**

**String midTitle=books.get(mid).getBookTitle();**

**int midId=books.get(mid).getBookId();**

**if(midTitle.equalsIgnoreCase(title)) {**

**System.*out*.println("Found!!-(Binary Search)");**

**books.get(mid).display();**

**break;**

**}else if(target < midId) {**

**left=mid+1;**

**}else {**

**right=mid-1;**

**}**

**}**

**}**

**}**

**public class Demo {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Scanner scan=new Scanner(System.*in*);**

**Book book1=new Book(1,"HarryPotter","james");**

**Book book2=new Book(2,"Alice in wonderland","William");**

**Search books=new Search();**

**books.add(book1);**

**books.add(book2);**

**System.*out*.println("Enter the book title :");**

**String title=scan.nextLine();**

**System.*out*.println("Linear search:");**

**books.linearSearch(title);**

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

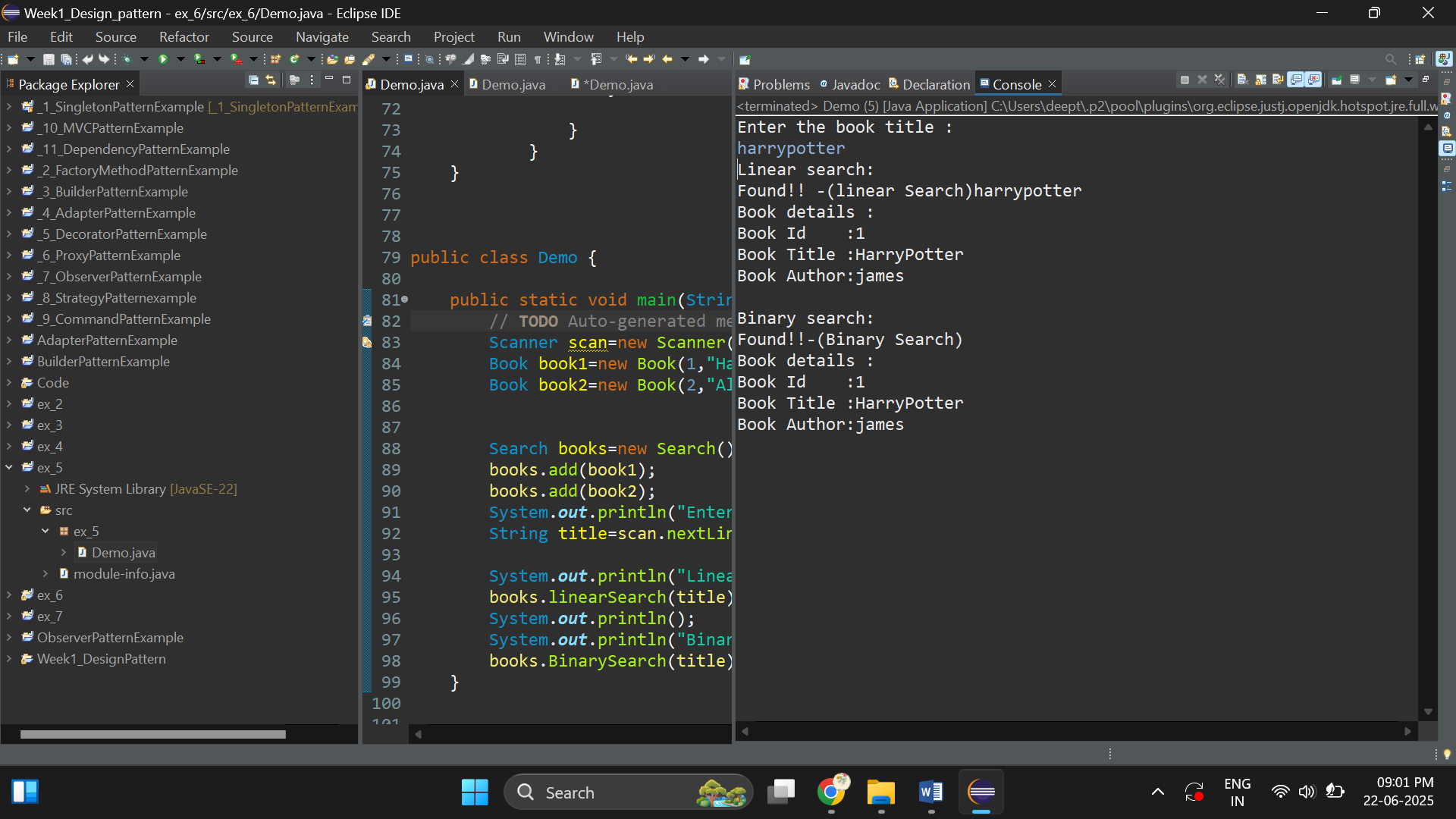
**System.*out*.println("Binary search:");**

**books.BinarySearch(title);**

**}**

**}**

**Output:**

****

**Exercise 7: Financial Forecasting**

**package ex\_7;**

**public class Demo {**

**// Recursive method to calculate future value**

**public static double forecastFutureValue(double presentValue, double growthRate, int years) {**

**//base condition**

**if (years == 0) {**

**return presentValue;**

**}**

**//recursive**

**return *forecastFutureValue*(presentValue \* (1 + growthRate), growthRate, years - 1);**

**}**

**public static void main(String[] args) {**

**double presentValue = 1000.0; // Starting money**

**double annualGrowthRate = 0.10; // 10% growth**

**int years = 5;**

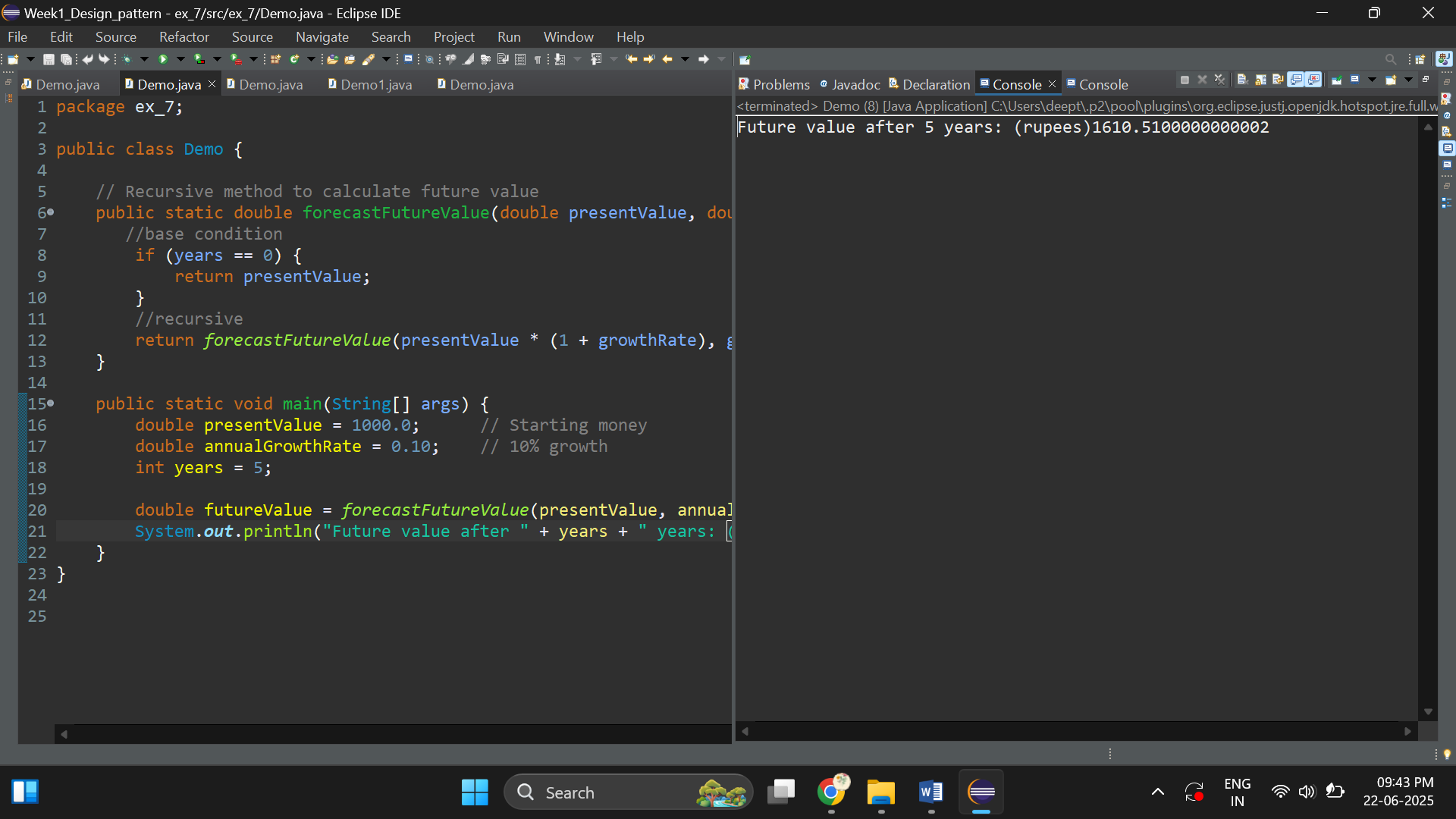
**double futureValue = *forecastFutureValue*(presentValue, annualGrowthRate, years);**

**System.*out*.println("Future value after " + years + " years: (rupees)" + futureValue);**

**}**

**}**

**Output:**

****

**­­**