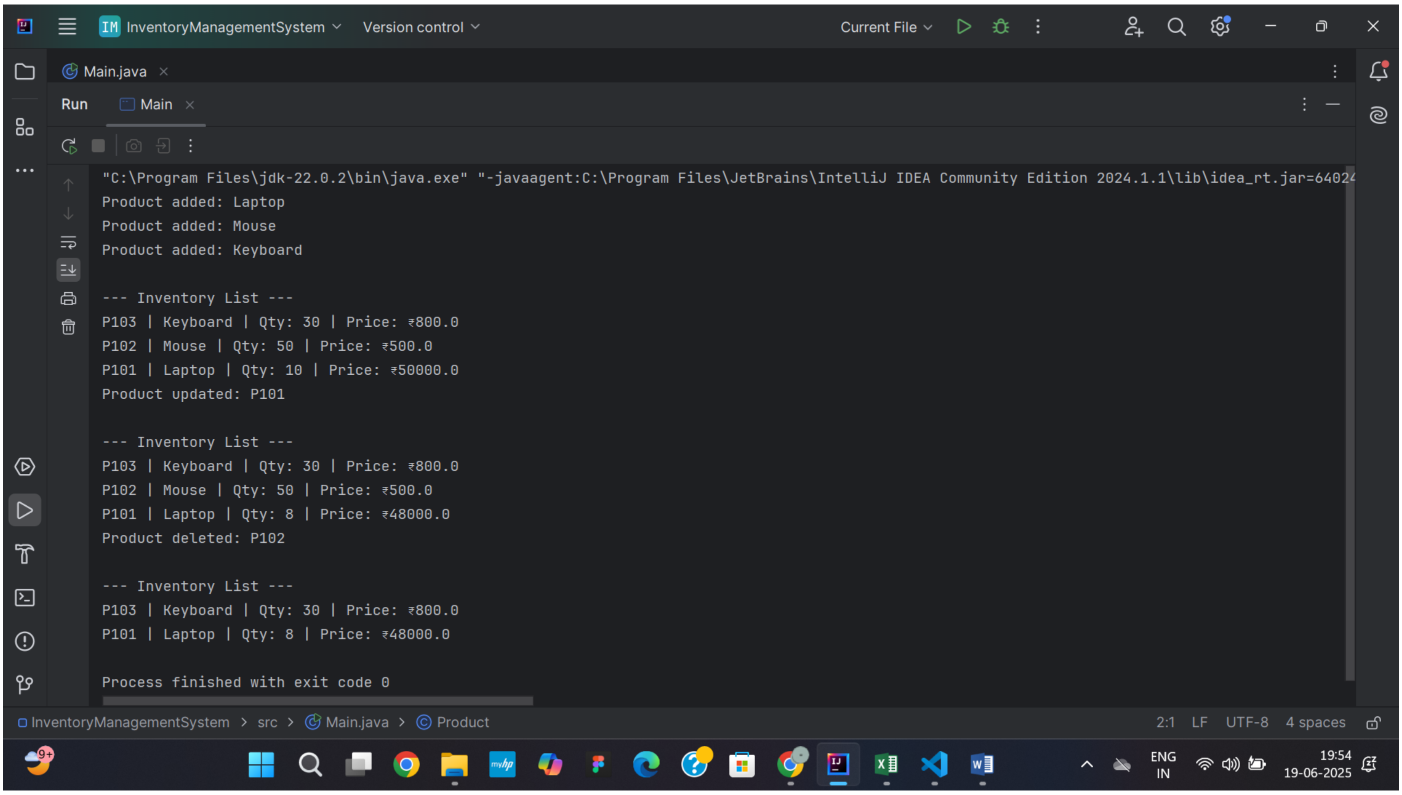
DATA STRUCTURES AND ALGORITHM

Exercise 1: Inventory Management System

```
import java.util.*;
class Product {
    String productId;
    String productName;
    int quantity;
    double price;

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

    public String toString() {
        return productId + " | " + productName + " | Qty: " + quantity + " | Price: ₹"
+ price;
    }
}

class InventoryManager {
    private Map<String, Product> inventory = new HashMap<>();

    public void addProduct(Product product) {
        if (inventory.containsKey(product.productId)) {
            System.out.println("Product with ID " + product.productId + " already
exists.");
        } else {
            inventory.put(product.productId, product);
            System.out.println("Product added: " + product.productName);
        }
    }

    public void updateProduct(String 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: " + productId);
        } else {
            System.out.println("Product not found: " + productId);
        }
    }

    public void deleteProduct(String productId) {
        if (inventory.remove(productId) != null) {
            System.out.println("Product deleted: " + productId);
        } else {
            System.out.println("Product not found: " + productId);
        }
    }

    public void displayAllProducts() {
        System.out.println("\n--- Inventory List ---");
        if (inventory.isEmpty()) {
            System.out.println("Inventory is empty.");
        } else {
            for (Product p : inventory.values()) {
                System.out.println(p);
            }
        }
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        InventoryManager manager = new InventoryManager();

        manager.addProduct(new Product("P101", "Laptop", 10, 50000));
        manager.addProduct(new Product("P102", "Mouse", 50, 500));
        manager.addProduct(new Product("P103", "Keyboard", 30, 800));

        manager.displayAllProducts();

        manager.updateProduct("P101", 8, 48000);

        manager.displayAllProducts();

        manager.deleteProduct("P102");

        manager.displayAllProducts();
    }
}
```

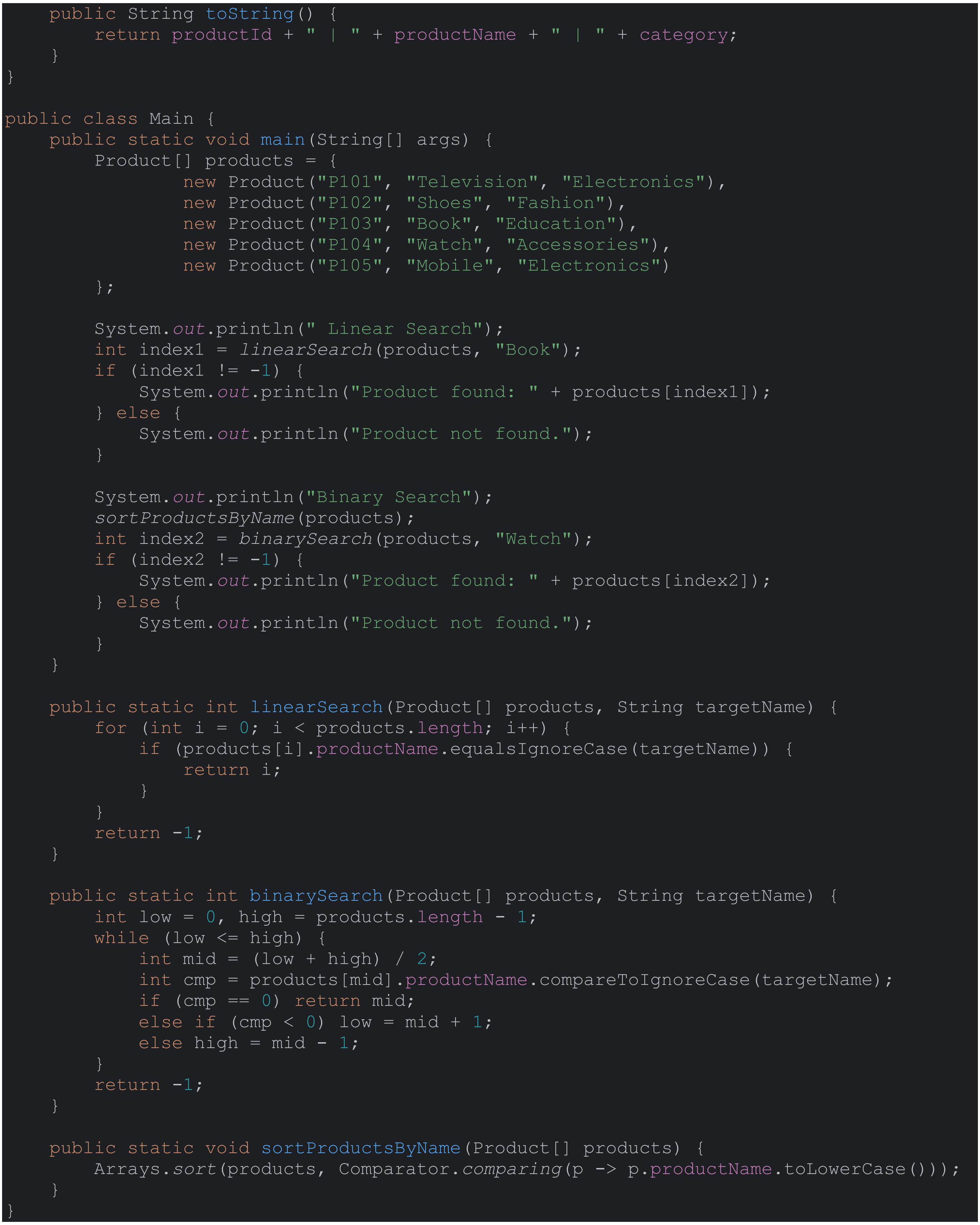
OUTPUT:



```
"C:\Program Files\jdk-22.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.1\lib\idea_rt.jar=64024
Product added: Laptop
Product added: Mouse
Product added: Keyboard

--- Inventory List ---
P103 | Keyboard | Qty: 30 | Price: ₹800.0
P102 | Mouse | Qty: 50 | Price: ₹500.0
P101 | Laptop | Qty: 10 | Price: ₹50000.0
Product updated: P101

--- Inventory List ---
P103 | Keyboard | Qty: 30 | Price: ₹800.0
P102 | Mouse | Qty: 50 | Price: ₹500.0
P101 | Laptop | Qty: 8 | Price: ₹48000.0
Product deleted: P102

--- Inventory List ---
P103 | Keyboard | Qty: 30 | Price: ₹800.0
P101 | Laptop | Qty: 8 | Price: ₹48000.0

Process finished with exit code 0
```

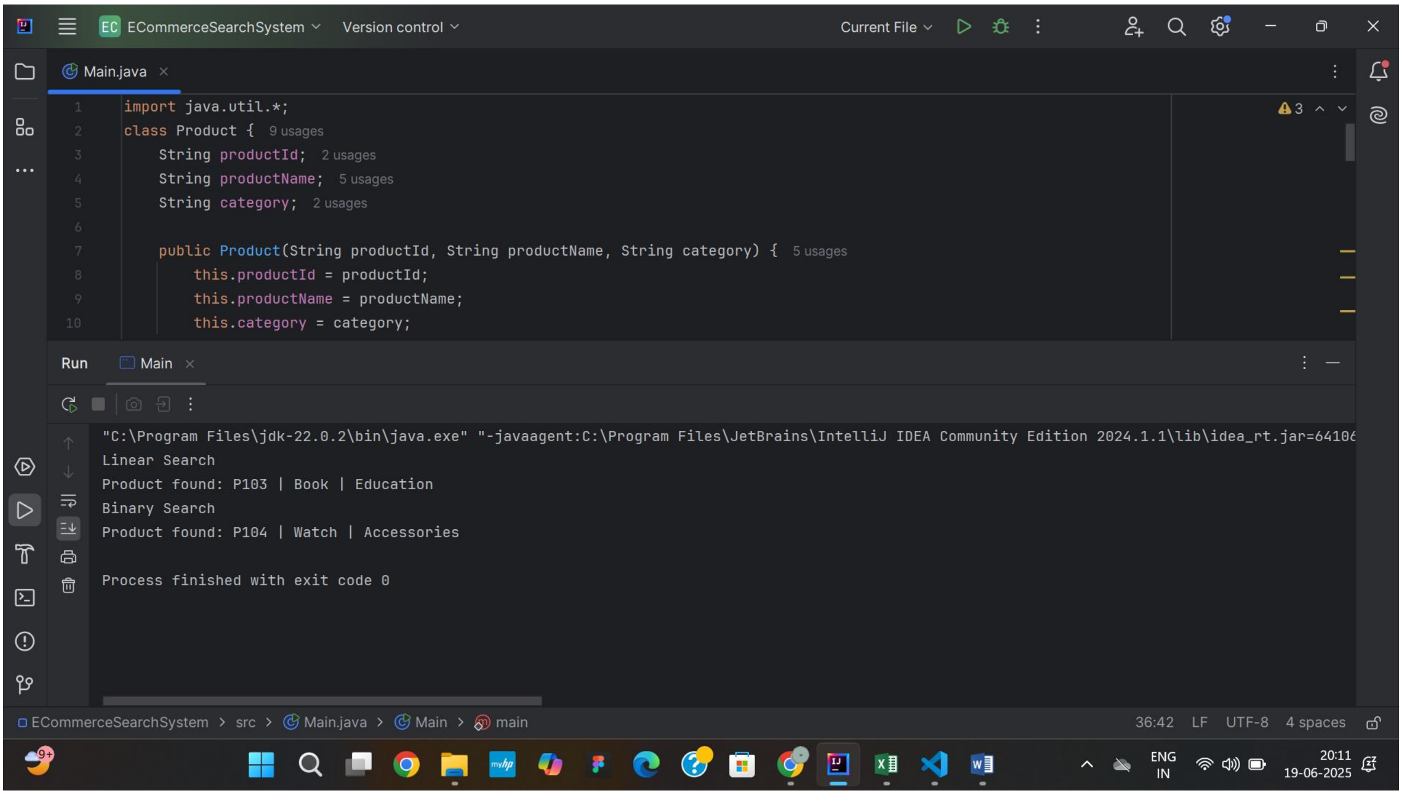
Exercise 2: E-commerce Platform Search Function

```
import java.util.*;
class Product {
    String productId;
    String productName;
    String category;

    public Product(String productId, String productName, String category) {
        this.productId = productId;
        this.productName = productName;
        this.category = category;
    }
```



```
    public String toString() {
        return productId + " | " + productName + " | " + category;
    }
}

public class Main {
    public static void main(String[] args) {
        Product[] products = {
                new Product("P101", "Television", "Electronics"),
                new Product("P102", "Shoes", "Fashion"),
                new Product("P103", "Book", "Education"),
                new Product("P104", "Watch", "Accessories"),
                new Product("P105", "Mobile", "Electronics")
        };

        System.out.println(" Linear Search");
        int index1 = linearSearch(products, "Book");
        if (index1 != -1) {
            System.out.println("Product found: " + products[index1]);
        } else {
            System.out.println("Product not found.");
        }

        System.out.println("Binary Search");
        sortProductsByName(products);
        int index2 = binarySearch(products, "Watch");
        if (index2 != -1) {
            System.out.println("Product found: " + products[index2]);
        } else {
            System.out.println("Product not found.");
        }
    }

    public static int linearSearch(Product[] products, String targetName) {
        for (int i = 0; i < products.length; i++) {
            if (products[i].productName.equalsIgnoreCase(targetName)) {
                return i;
            }
        }
        return -1;
    }

    public static int binarySearch(Product[] products, String targetName) {
        int low = 0, high = products.length - 1;
        while (low <= high) {
            int mid = (low + high) / 2;
            int cmp = products[mid].productName.compareToIgnoreCase(targetName);
            if (cmp == 0) return mid;
            else if (cmp < 0) low = mid + 1;
            else high = mid - 1;
        }
        return -1;
    }

    public static void sortProductsByName(Product[] products) {
        Arrays.sort(products, Comparator.comparing(p -> p.productName.toLowerCase()));
    }
}
```

OUTPUT:

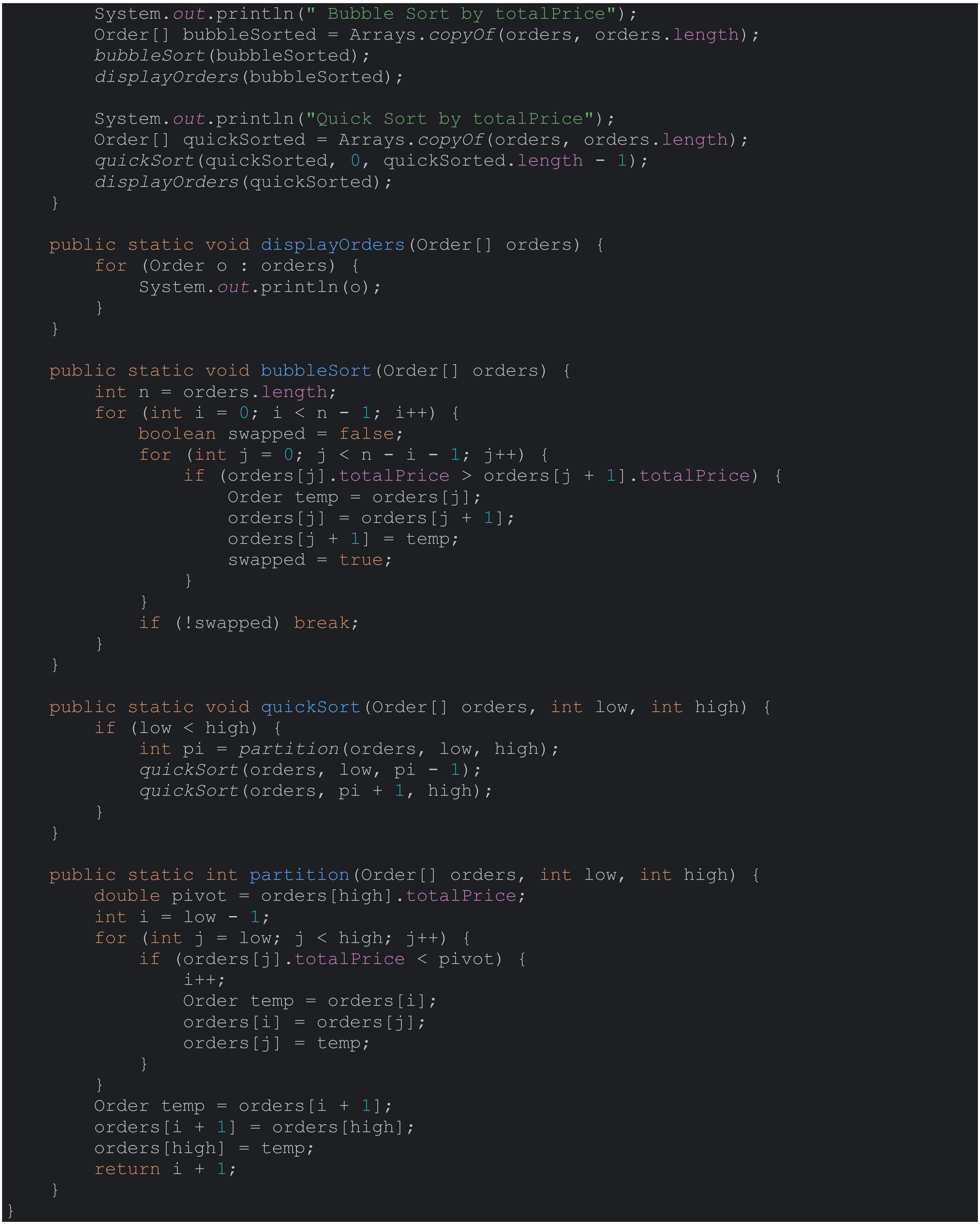


```java
import java.util.*;
class Product {
    String productId;
    String productName;
    String category;

    public Product(String productId, String productName, String category) {
        this.productId = productId;
        this.productName = productName;
        this.category = category;
```

```
"C:\Program Files\jdk-22.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.1\lib\idea_rt.jar=6410
Linear Search
Product found: P103 | Book | Education
Binary Search
Product found: P104 | Watch | Accessories

Process finished with exit code 0
```

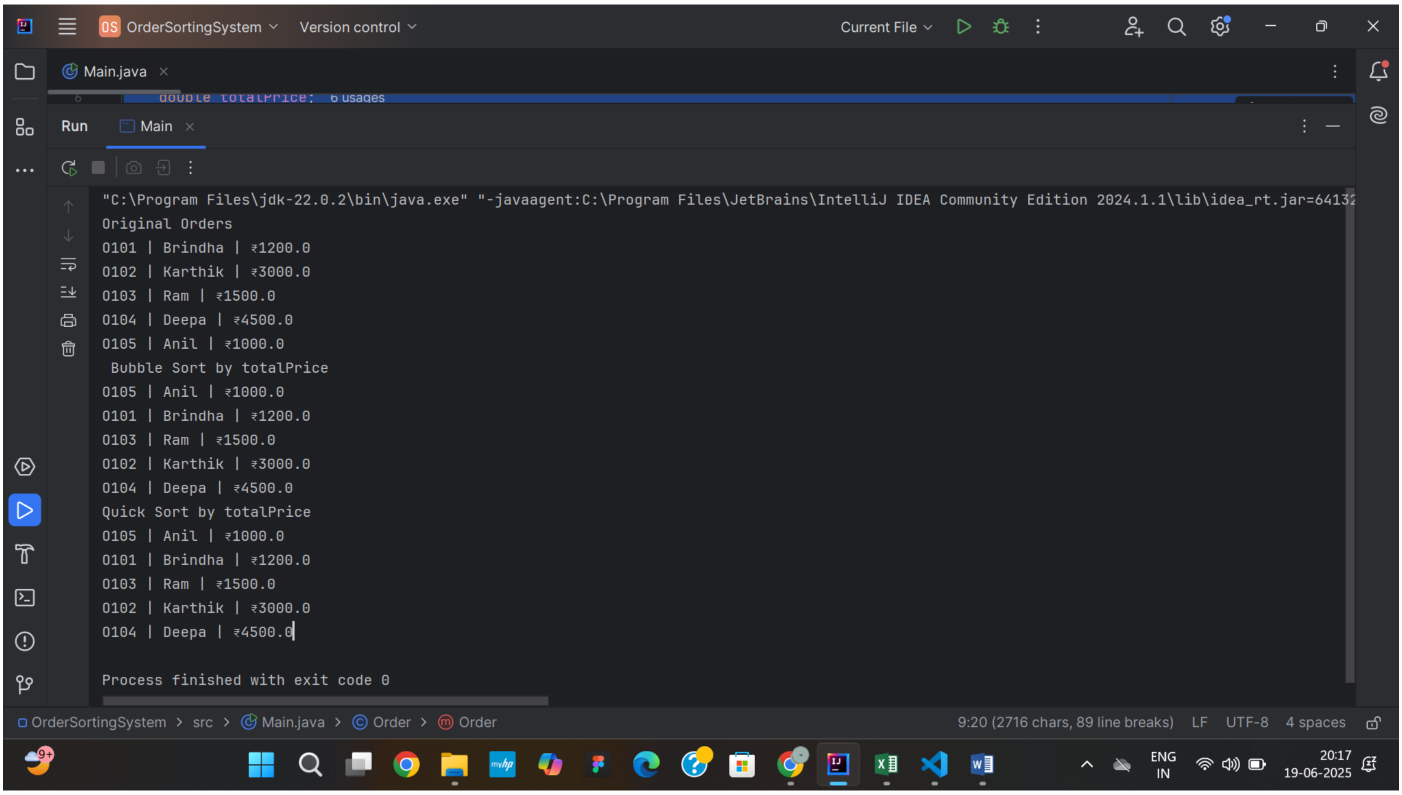
Exercise 3: Sorting Customer Orders

```java
import java.util.*;

class Order {
    String orderId;
    String customerName;
    double totalPrice;

    public Order(String orderId, String customerName, double totalPrice) {
        this.orderId = orderId;
        this.customerName = customerName;
        this.totalPrice = totalPrice;
    }

    public String toString() {
        return orderId + " | " + customerName + " | ₹" + totalPrice;
    }
}

public class Main {
    public static void main(String[] args) {
        Order[] orders = {
                new Order("O101", "Brindha", 1200),
                new Order("O102", "Karthik", 3000),
                new Order("O103", "Ram", 1500),
                new Order("O104", "Deepa", 4500),
                new Order("O105", "Anil", 1000)
        };

        System.out.println("Original Orders");
        displayOrders(orders);
```



```java
        System.out.println(" Bubble Sort by totalPrice");
        Order[] bubbleSorted = Arrays.copyOf(orders, orders.length);
        bubbleSort(bubbleSorted);
        displayOrders(bubbleSorted);

        System.out.println("Quick Sort by totalPrice");
        Order[] quickSorted = Arrays.copyOf(orders, orders.length);
        quickSort(quickSorted, 0, quickSorted.length - 1);
        displayOrders(quickSorted);
    }

    public static void displayOrders(Order[] orders) {
        for (Order o : orders) {
            System.out.println(o);
        }
    }

    public static void bubbleSort(Order[] orders) {
        int n = orders.length;
        for (int i = 0; i < n - 1; i++) {
            boolean swapped = false;
            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;
                    swapped = true;
                }
            }
            if (!swapped) break;
        }
    }

    public static void quickSort(Order[] orders, int low, int high) {
        if (low < high) {
            int pi = partition(orders, low, high);
            quickSort(orders, low, pi - 1);
            quickSort(orders, pi + 1, high);
        }
    }

    public 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;
    }
}
```

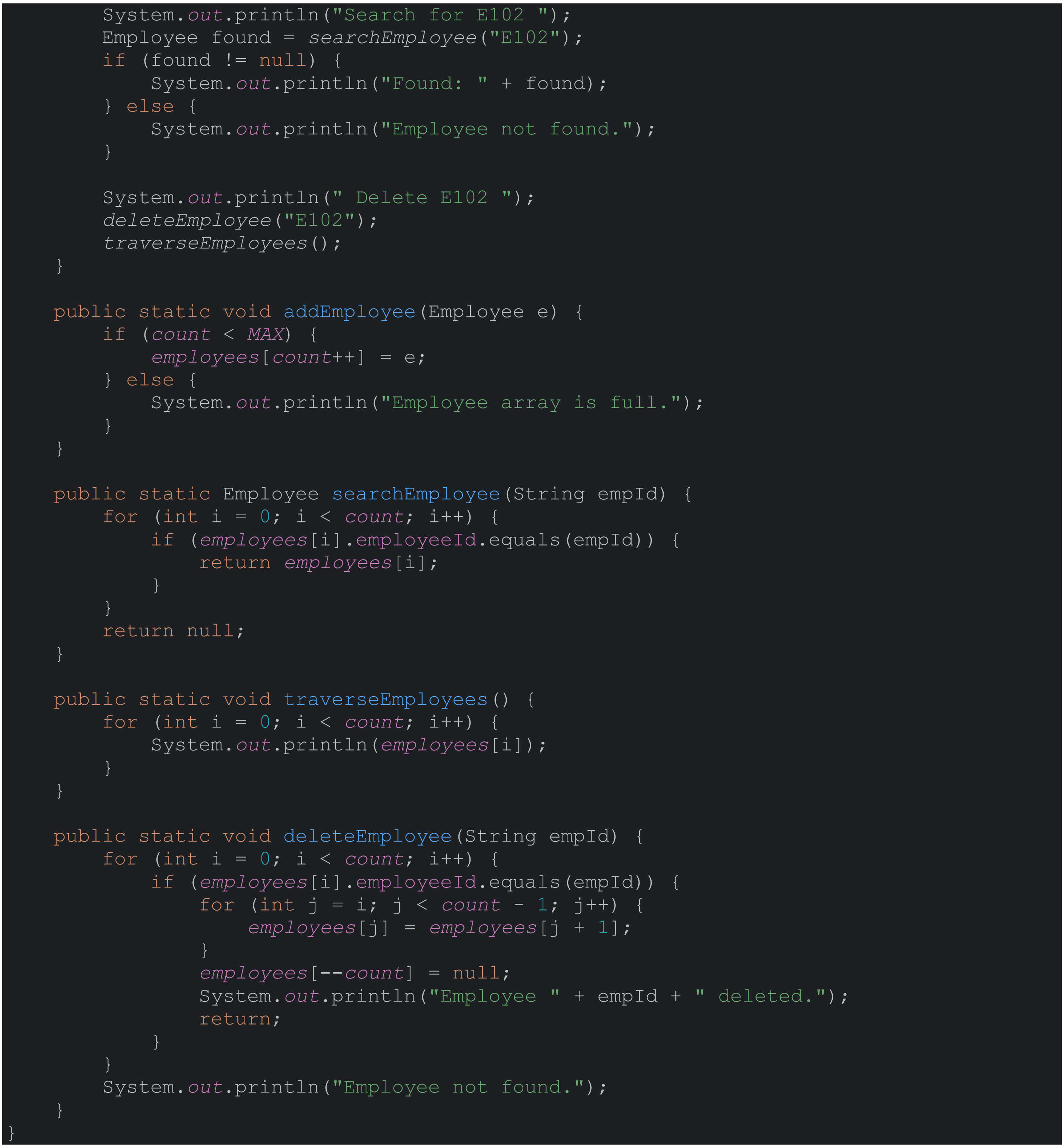
OUTPUT:


OrderSortingSystem
Version control
Current File
Main.java
Run
Main
OrderSortingSystem > src > Main.java > Order > Order
9:20 (2716 chars, 89 line breaks) LF UTF-8 4 spaces
ENG IN
20:17
19-06-2025

```
"C:\Program Files\jdk-22.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.1\lib\idea_rt.jar=64132
Original Orders
0101 | Brindha | ₹1200.0
0102 | Karthik | ₹3000.0
0103 | Ram | ₹1500.0
0104 | Deepa | ₹4500.0
0105 | Anil | ₹1000.0
 Bubble Sort by totalPrice
0105 | Anil | ₹1000.0
0101 | Brindha | ₹1200.0
0103 | Ram | ₹1500.0
0102 | Karthik | ₹3000.0
0104 | Deepa | ₹4500.0
Quick Sort by totalPrice
0105 | Anil | ₹1000.0
0101 | Brindha | ₹1200.0
0103 | Ram | ₹1500.0
0102 | Karthik | ₹3000.0
0104 | Deepa | ₹4500.0

Process finished with exit code 0
```

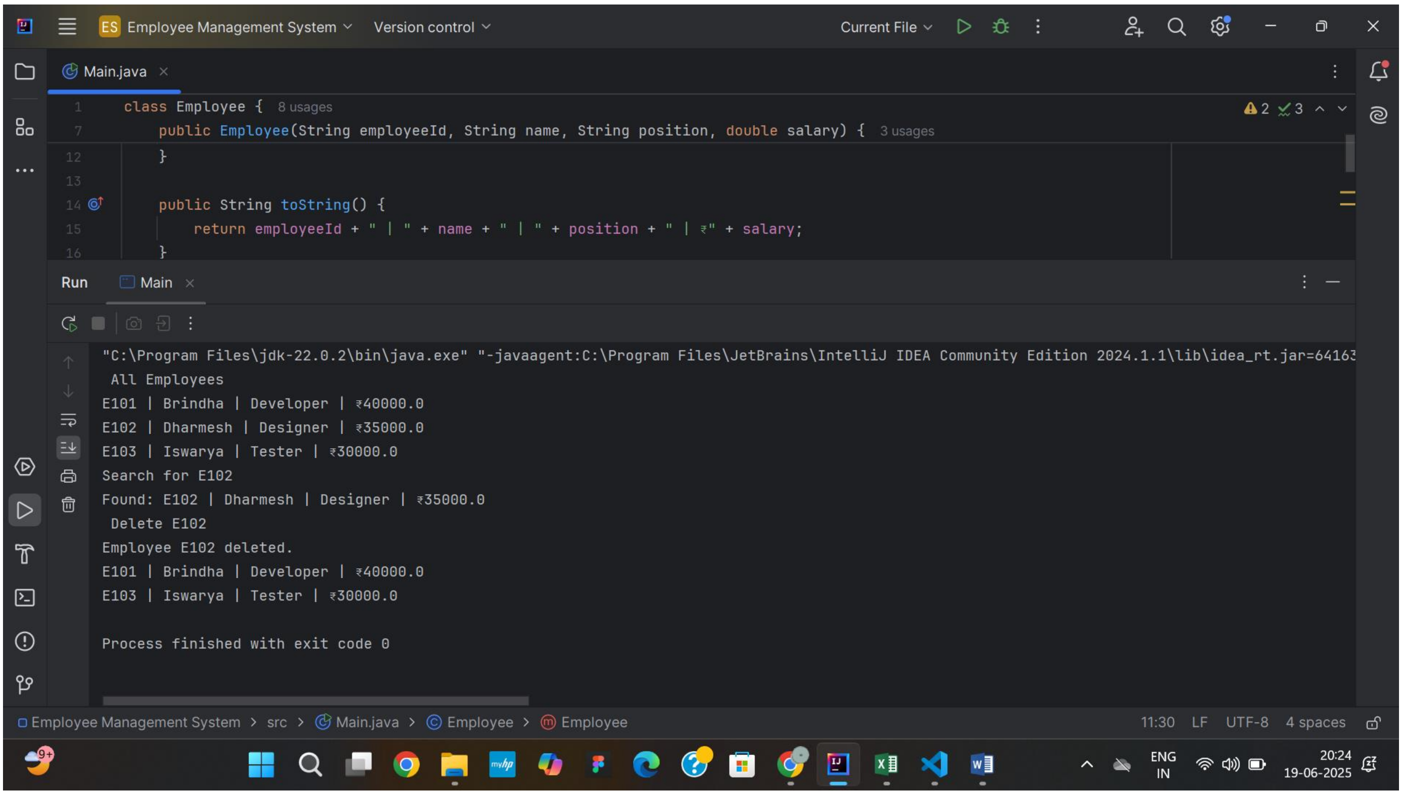
Exercise 4: Employee Management System

```java
class Employee {
    String employeeId;
    String name;
    String position;
    double salary;

    public Employee(String employeeId, String name, String position, double salary) {
        this.employeeId = employeeId;
        this.name = name;
        this.position = position;
        this.salary = salary;
    }

    public String toString() {
        return employeeId + " | " + name + " | " + position + " | ₹" + salary;
    }
}

public class Main {
    static final int MAX = 100;
    static Employee[] employees = new Employee[MAX];
    static int count = 0;

    public static void main(String[] args) {
        addEmployee(new Employee("E101", "Brindha", "Developer", 40000));
        addEmployee(new Employee("E102", "Dharmesh", "Designer", 35000));
        addEmployee(new Employee("E103", "Iswarya", "Tester", 30000));

        System.out.println(" All Employees");
        traverseEmployees();
```



```
        System.out.println("Search for E102 ");
        Employee found = searchEmployee("E102");
        if (found != null) {
            System.out.println("Found: " + found);
        } else {
            System.out.println("Employee not found.");
        }

        System.out.println(" Delete E102 ");
        deleteEmployee("E102");
        traverseEmployees();
    }

    public static void addEmployee(Employee e) {
        if (count < MAX) {
            employees[count++] = e;
        } else {
            System.out.println("Employee array is full.");
        }
    }

    public static Employee searchEmployee(String empId) {
        for (int i = 0; i < count; i++) {
            if (employees[i].employeeId.equals(empId)) {
                return employees[i];
            }
        }
        return null;
    }

    public static void traverseEmployees() {
        for (int i = 0; i < count; i++) {
            System.out.println(employees[i]);
        }
    }

    public static void deleteEmployee(String empId) {
        for (int i = 0; i < count; i++) {
            if (employees[i].employeeId.equals(empId)) {
                for (int j = i; j < count - 1; j++) {
                    employees[j] = employees[j + 1];
                }
                employees[--count] = null;
                System.out.println("Employee " + empId + " deleted.");
                return;
            }
        }
        System.out.println("Employee not found.");
    }
}
```

OUTPUT:



Employee Management System
Version control
Current File
Main.java
Run
Main
Employee Management System > src > Main.java > Employee > Employee
11:30 LF UTF-8 4 spaces
ENG IN
20:24
19-06-2025

```
class Employee {  8 usages
    public Employee(String employeeId, String name, String position, double salary) {  3 usages
    }

    public String toString() {
        return employeeId + " | " + name + " | " + position + " | ₹" + salary;
    }
```

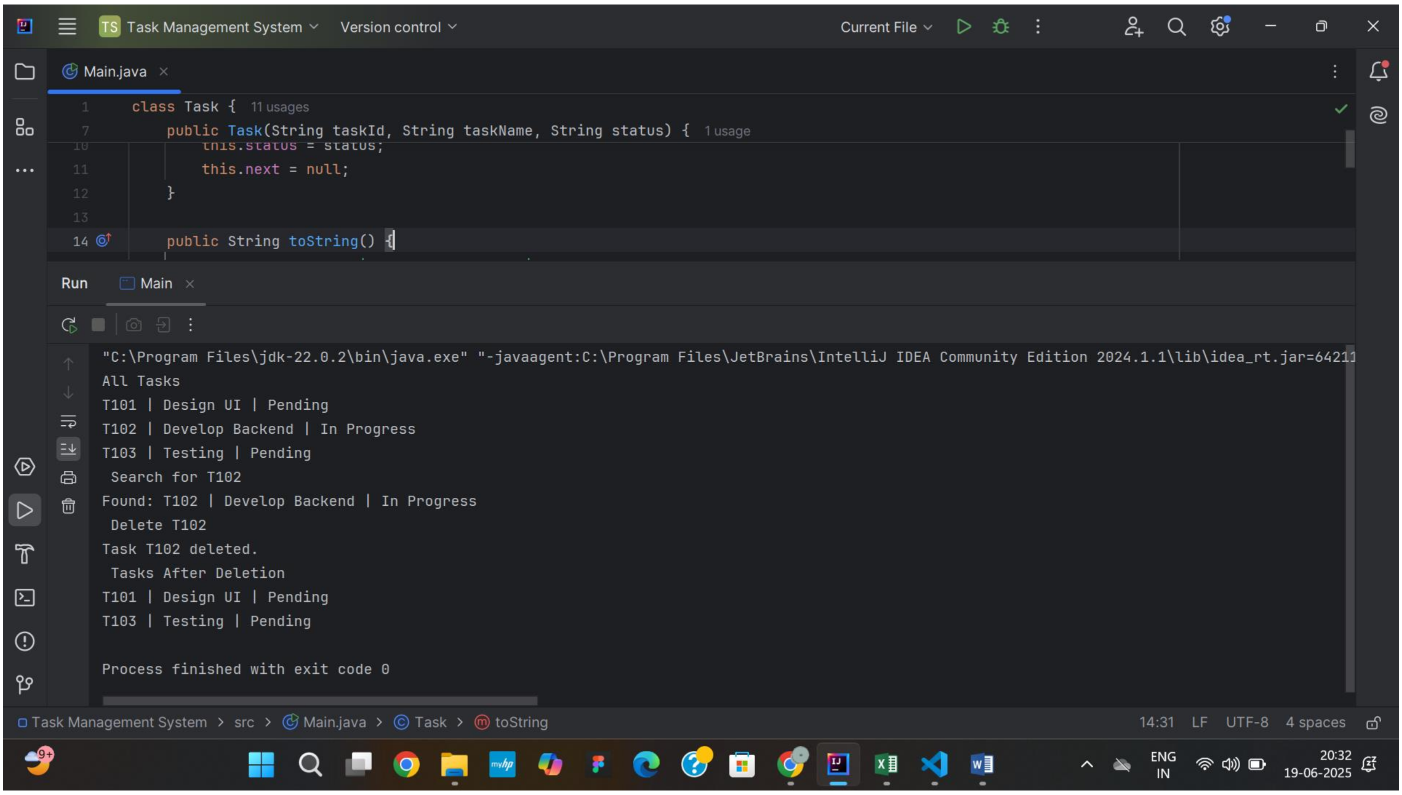
```
"C:\Program Files\jdk-22.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.1\lib\idea_rt.jar=64163
 All Employees
E101 | Brindha | Developer | ₹40000.0
E102 | Dharmesh | Designer | ₹35000.0
E103 | Iswarya | Tester | ₹30000.0
Search for E102
Found: E102 | Dharmesh | Designer | ₹35000.0
 Delete E102
Employee E102 deleted.
E101 | Brindha | Developer | ₹40000.0
E103 | Iswarya | Tester | ₹30000.0

Process finished with exit code 0
```

Exercise 5: Task Management System

```
class Task {
    String taskId;
    String taskName;
    String status;
    Task next;

    public Task(String taskId, String taskName, String status) {
        this.taskId = taskId;
        this.taskName = taskName;
        this.status = status;
        this.next = null;
    }

    public String toString() {
        return taskId + " | " + taskName + " | " + status;
    }
}

class TaskList {
    Task head;

    public void addTask(String taskId, String taskName, String status) {
        Task newTask = new Task(taskId, taskName, status);
        if (head == null) {
            head = newTask;
        } else {
            Task temp = head;
            while (temp.next != null) {
                temp = temp.next;
            }
            temp.next = newTask;
```

```java
        }
    }

    public Task searchTask(String taskId) {
        Task temp = head;
        while (temp != null) {
            if (temp.taskId.equals(taskId)) {
                return temp;
            }
            temp = temp.next;
        }
        return null;
    }

    public void deleteTask(String taskId) {
        if (head == null) {
            System.out.println("Task list is empty.");
            return;
        }

        if (head.taskId.equals(taskId)) {
            head = head.next;
            System.out.println("Task " + taskId + " deleted.");
            return;
        }

        Task prev = head;
        Task current = head.next;

        while (current != null) {
            if (current.taskId.equals(taskId)) {
                prev.next = current.next;
                System.out.println("Task " + taskId + " deleted.");
                return;
            }
            prev = current;
            current = current.next;
        }

        System.out.println("Task not found.");
    }

    public void traverseTasks() {
        if (head == null) {
            System.out.println("No tasks available.");
            return;
        }
        Task temp = head;
        while (temp != null) {
            System.out.println(temp);
            temp = temp.next;
        }
    }
}

public class Main {
    public static void main(String[] args) {
        TaskList taskList = new TaskList();

        taskList.addTask("T101", "Design UI", "Pending");
        taskList.addTask("T102", "Develop Backend", "In Progress");
        taskList.addTask("T103", "Testing", "Pending");

        System.out.println("All Tasks ");
        taskList.traverseTasks();

        System.out.println(" Search for T102 ");
        Task found = taskList.searchTask("T102");
        System.out.println(found != null ? "Found: " + found : "Task not found.");

        System.out.println(" Delete T102 ");
        taskList.deleteTask("T102");

        System.out.println(" Tasks After Deletion ");
        taskList.traverseTasks();
    }
}
```

OUTPUT:

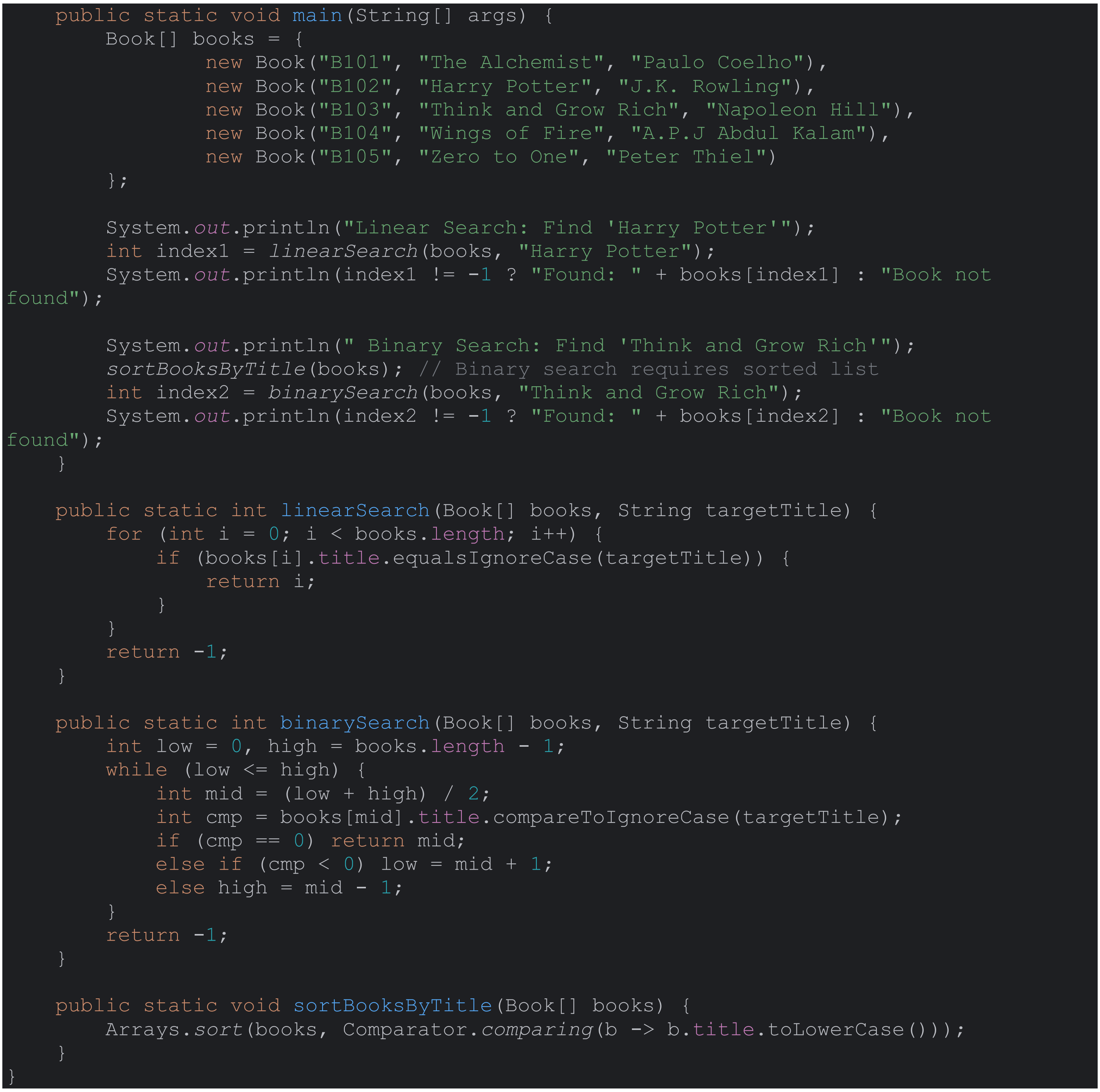


```
class Task {
    public Task(String taskId, String taskName, String status) {
        this.status = status;
        this.next = null;
    }

    public String toString() {
```

```
"C:\Program Files\jdk-22.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.1\lib\idea_rt.jar=64211
All Tasks
T101 | Design UI | Pending
T102 | Develop Backend | In Progress
T103 | Testing | Pending
 Search for T102
Found: T102 | Develop Backend | In Progress
 Delete T102
Task T102 deleted.
 Tasks After Deletion
T101 | Design UI | Pending
T103 | Testing | Pending

Process finished with exit code 0
```

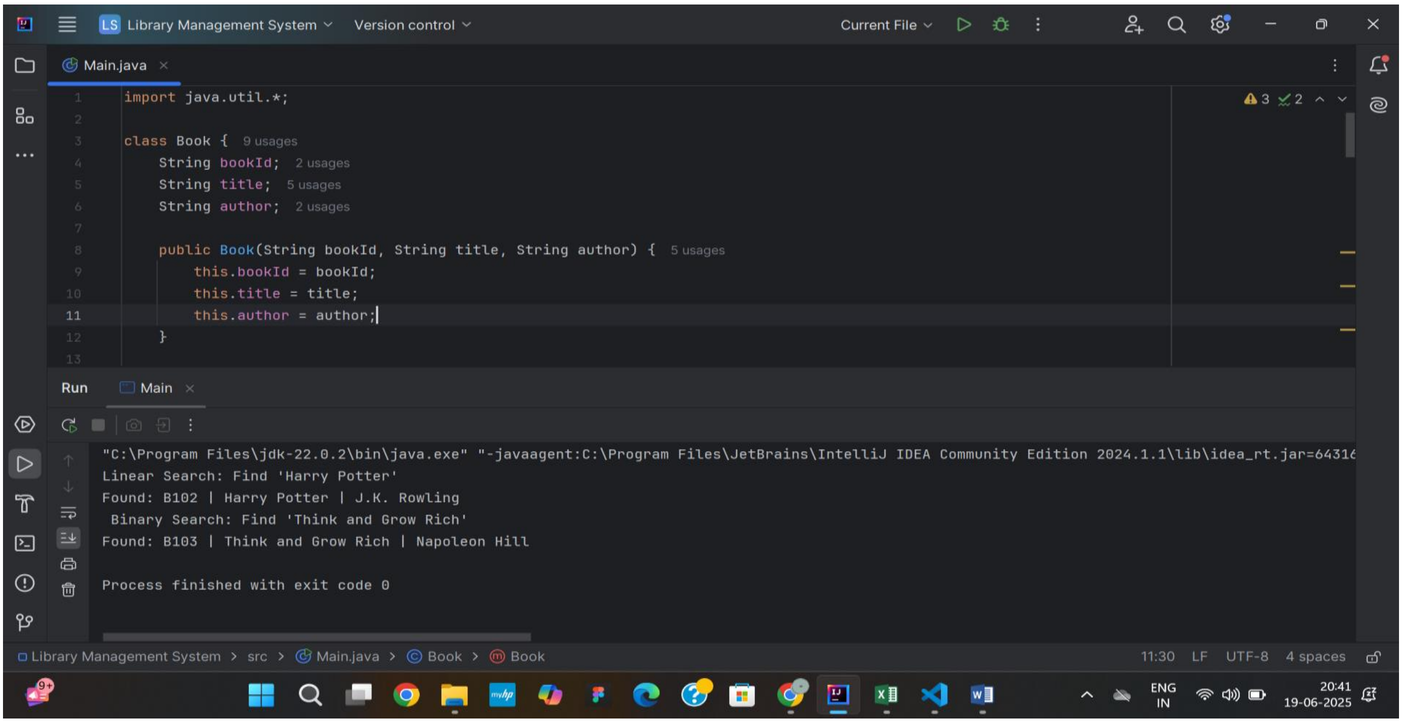
Exercise 6: Library Management System

```
import java.util.*;

class Book {
    String bookId;
    String title;
    String author;

    public Book(String bookId, String title, String author) {
        this.bookId = bookId;
        this.title = title;
        this.author = author;
    }

    public String toString() {
        return bookId + " | " + title + " | " + author;
    }
}

public class Main {
```



```java
    public static void main(String[] args) {
        Book[] books = {
                new Book("B101", "The Alchemist", "Paulo Coelho"),
                new Book("B102", "Harry Potter", "J.K. Rowling"),
                new Book("B103", "Think and Grow Rich", "Napoleon Hill"),
                new Book("B104", "Wings of Fire", "A.P.J Abdul Kalam"),
                new Book("B105", "Zero to One", "Peter Thiel")
        };

        System.out.println("Linear Search: Find 'Harry Potter'");
        int index1 = linearSearch(books, "Harry Potter");
        System.out.println(index1 != -1 ? "Found: " + books[index1] : "Book not
found");

        System.out.println(" Binary Search: Find 'Think and Grow Rich'");
        sortBooksByTitle(books); // Binary search requires sorted list
        int index2 = binarySearch(books, "Think and Grow Rich");
        System.out.println(index2 != -1 ? "Found: " + books[index2] : "Book not
found");
    }

    public static int linearSearch(Book[] books, String targetTitle) {
        for (int i = 0; i < books.length; i++) {
            if (books[i].title.equalsIgnoreCase(targetTitle)) {
                return i;
            }
        }
        return -1;
    }

    public static int binarySearch(Book[] books, String targetTitle) {
        int low = 0, high = books.length - 1;
        while (low <= high) {
            int mid = (low + high) / 2;
            int cmp = books[mid].title.compareToIgnoreCase(targetTitle);
            if (cmp == 0) return mid;
            else if (cmp < 0) low = mid + 1;
            else high = mid - 1;
        }
        return -1;
    }

    public static void sortBooksByTitle(Book[] books) {
        Arrays.sort(books, Comparator.comparing(b -> b.title.toLowerCase()));
    }
}
```

OUTPUT:



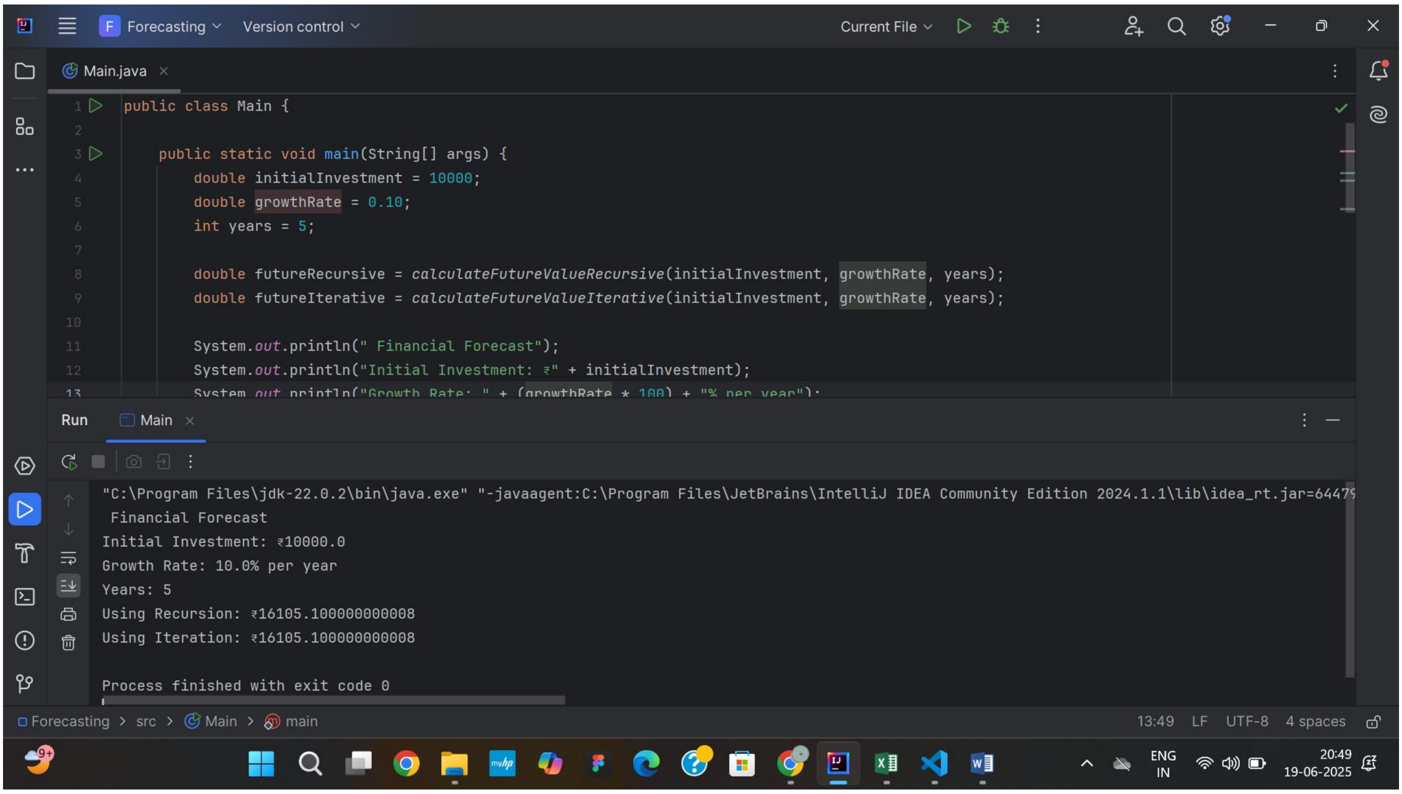
```java
import java.util.*;

class Book {
    String bookId;
    String title;
    String author;

    public Book(String bookId, String title, String author) {
        this.bookId = bookId;
        this.title = title;
        this.author = author;
    }

```

```
"C:\Program Files\jdk-22.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.1\lib\idea_rt.jar=64316
Linear Search: Find 'Harry Potter'
Found: B102 | Harry Potter | J.K. Rowling
 Binary Search: Find 'Think and Grow Rich'
Found: B103 | Think and Grow Rich | Napoleon Hill

Process finished with exit code 0
```

Exercise 7: Financial Forecasting

```java
public class Main {

    public static void main(String[] args) {
        double initialInvestment = 10000;
        double growthRate = 0.10;
        int years = 5;

        double futureRecursive = calculateFutureValueRecursive(initialInvestment,
growthRate, years);
        double futureIterative = calculateFutureValueIterative(initialInvestment,
growthRate, years);

        System.out.println(" Financial Forecast");
        System.out.println("Initial Investment: ₹" + initialInvestment);
        System.out.println("Growth Rate: " + (growthRate * 100) + "% per year");
        System.out.println("Years: " + years);
        System.out.println("Using Recursion: ₹" + futureRecursive);
        System.out.println("Using Iteration: ₹" + futureIterative);
    }

    public static double calculateFutureValueRecursive(double currentValue, double
growthRate, int years) {
        if (years == 0) {
            return currentValue;
        }
        return calculateFutureValueRecursive(currentValue, growthRate, years - 1) * (1
+ growthRate);
    }

    public static double calculateFutureValueIterative(double currentValue, double
growthRate, int years) {
        for (int i = 0; i < years; i++) {
            currentValue *= (1 + growthRate);
        }
        return currentValue;
    }
}
```

OUTPUT:



```
public class Main {

    public static void main(String[] args) {
        double initialInvestment = 10000;
        double growthRate = 0.10;
        int years = 5;

        double futureRecursive = calculateFutureValueRecursive(initialInvestment, growthRate, years);
        double futureIterative = calculateFutureValueIterative(initialInvestment, growthRate, years);

        System.out.println(" Financial Forecast");
        System.out.println("Initial Investment: ₹" + initialInvestment);
        System.out.println("Growth Rate: " + (growthRate * 100) + "% per year");
```

```
"C:\Program Files\jdk-22.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.1\lib\idea_rt.jar=6447
 Financial Forecast
Initial Investment: ₹10000.0
Growth Rate: 10.0% per year
Years: 5
Using Recursion: ₹16105.100000000008
Using Iteration: ₹16105.100000000008

Process finished with exit code 0
```