Data Structures and Algorithms

Exercise 3:

Sorting Customer Orders

This project implements sorting of customer orders on an e-commerce platform using Java. It compares Bubble Sort and Quick Sort algorithms to efficiently prioritize high-value orders. Here’s a detailed explanation:

Step 1: Understand Sorting Algorithms

- **Bubble Sort:** Simple but inefficient for large data sets – time complexity *$O(n^2)$*
- **Quick Sort:** Efficient and widely used – average case *$O(n \log n)$*, worst case *$O(n^2)$*

Step 2: Setup

- *Order* class with *orderId*, *customerName*, and *totalPrice*.

Step 3: Implementation

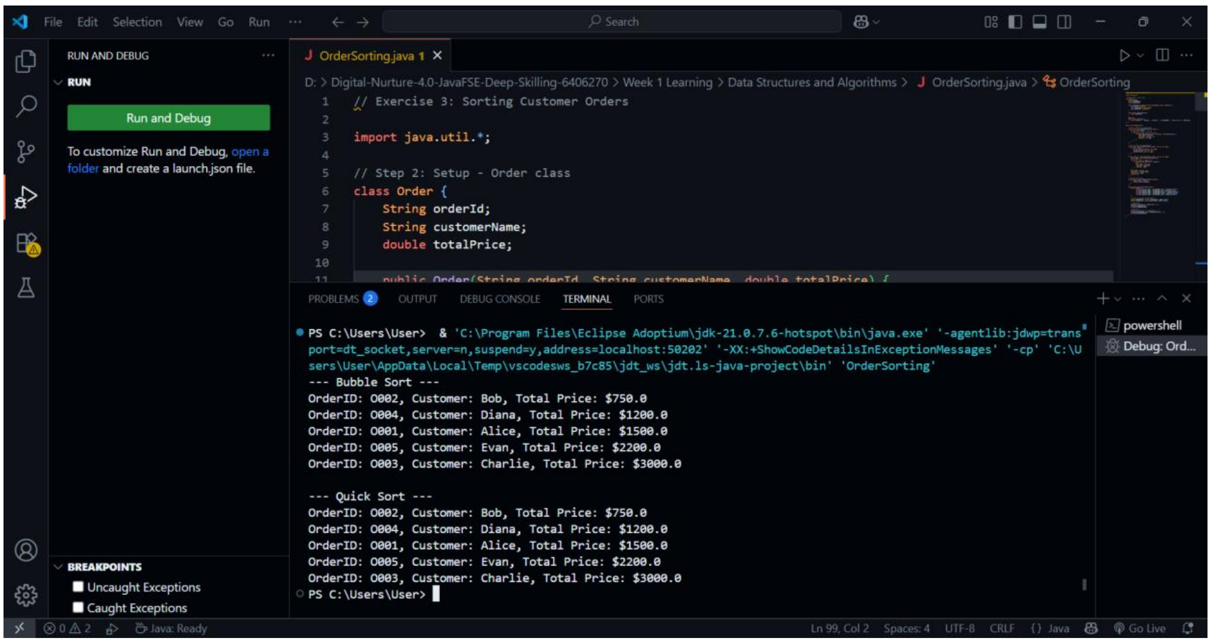
- *bubbleSort()* sorts using nested loops.
- *quickSort()* uses divide-and-conquer strategy with partitioning.

Step 4: Analysis

Algorithm	Best Case	Average Case	Worst Case
Bubble Sort	$O(n)$	$O(n^2)$	$O(n^2)$
Quick Sort	$O(n \log n)$	$O(n \log n)$	$O(n^2)$

- **Quick Sort** is generally preferred due to better average-case performance and scalability.

Output



File Edit Selection View Go Run
Search
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug, open a folder and create a launch.json file.
BREAKPOINTS
Uncaught Exceptions
Caught Exceptions
OrderSorting.java 1
D: > Digital-Nurture-4.0-JavaFSE-Deep-Skilling-6406270 > Week 1 Learning > Data Structures and Algorithms > OrderSorting.java > OrderSorting
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
Debug: Ord...
Ln 99, Col 2 Spaces: 4 UTF-8 CRLF {} Java Go Live
Java: Ready

```java
// Exercise 3: Sorting Customer Orders

import java.util.*;

// Step 2: Setup - Order class
class Order {
    String orderId;
    String customerName;
    double totalPrice;

```

```
PS C:\Users\User>  & 'C:\Program Files\Eclipse Adoptium\jdk-21.0.7.6-hotspot\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:50202' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\User\AppData\Local\Temp\vscodesws_b7c85\jdt_ws\jdt.ls-java-project\bin' 'OrderSorting'
--- Bubble Sort ---
OrderID: O002, Customer: Bob, Total Price: $750.0
OrderID: O004, Customer: Diana, Total Price: $1200.0
OrderID: O001, Customer: Alice, Total Price: $1500.0
OrderID: O005, Customer: Evan, Total Price: $2200.0
OrderID: O003, Customer: Charlie, Total Price: $3000.0

--- Quick Sort ---
OrderID: O002, Customer: Bob, Total Price: $750.0
OrderID: O004, Customer: Diana, Total Price: $1200.0
OrderID: O001, Customer: Alice, Total Price: $1500.0
OrderID: O005, Customer: Evan, Total Price: $2200.0
OrderID: O003, Customer: Charlie, Total Price: $3000.0
PS C:\Users\User>
```