**Data Structures - Hands-On**

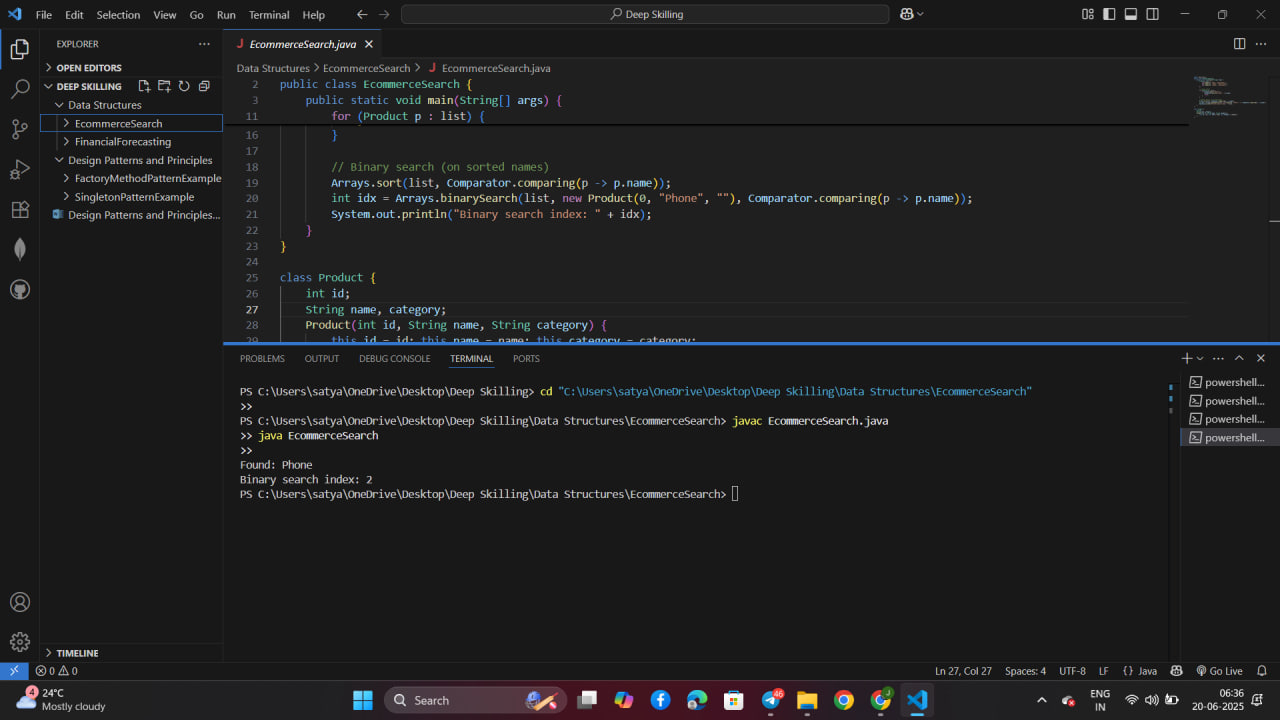
**Exercise 2: E-commerce Platform Search Function**

***→ Java File: EcommerceSearch.java***

import java.util.\*;  
public class EcommerceSearch {  
 public static void main(String[] args) {  
 Product[] list = {  
 new Product(1, "Pen", "Stationery"),  
 new Product(2, "Phone", "Electronics"),  
 new Product(3, "Paper", "Stationery")  
 };  
  
 // Linear search  
 for (Product p : list) {  
 if (p.name.equals("Phone")) {  
 System.out.println("Found: " + p.name);  
 break;  
 }  
 }  
  
 // Binary search (on sorted names)  
 Arrays.sort(list, Comparator.comparing(p -> p.name));  
 int idx = Arrays.binarySearch(list, new Product(0, "Phone", ""), Comparator.comparing(p -> p.name));  
 System.out.println("Binary search index: " + idx);  
 }  
}  
  
class Product {  
 int id;  
 String name, category;  
 Product(int id, String name, String category) {  
 this.id = id; this.name = name; this.category = category;  
 }  
}

Output Preview:

Found: Phone  
Binary search index: 2

**Exercise 7: Financial Forecasting**

***→ Java File: FinancialForecasting.java***

public class FinancialForecasting {  
 public static void main(String[] args) {  
 double result = predict(1000, 0.10, 5);  
 System.out.println("Future: " + result);  
 }  
 public static double predict(double amount, double rate, int years) {  
 if (years == 0) return amount;  
 return predict(amount \* (1 + rate), rate, years - 1);  
 }  
}

Output Preview:

Future: 1610.5100000000002

