**Week One: Data Structures & Algorithms  
  
Exercise 2: E-commerce Platform Search Function  
  
Scenario:**

Develop a product search feature for an e-commerce platform where users should be able to search by product name using Linear Search for small lists and Binary Search (with sorting) for faster results on larger catalogs. **CODE:**  
  
**Class Product:**

public class Product {

int productId;

String productName;

String category;

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

this.productId = productId;

this.productName = productName;

this.category = category;

}

public String toString() {

return productId + ": " + productName + " [" + category + "]";

}

}

**Class LinearSearch:**

public class LinearSearch {

public static Product search(Product[] list, String name) {

for (Product p : list) {

if (p.productName.equalsIgnoreCase(name)) {

return p;

}

}

return null;

}

}

**Class BinarySearch:**

import java.util.Arrays;

public class BinarySearch {

public static void sortProducts(Product[] products) {

Arrays.sort(products, (a, b) -> a.productName.compareToIgnoreCase(b.productName));

}

public static Product search(Product[] products, String name) {

int left = 0;

int right = products.length - 1;

while (left <= right) {

int mid = (left + right) / 2;

int compare = products[mid].productName.compareToIgnoreCase(name);

if (compare == 0) return products[mid];

else if (compare < 0) left = mid + 1;

else right = mid - 1;

}

return null;

}

}

**Class: SearchComparison**

public class SearchComparison {

public static void main(String[] args) {

Product[] catalog = {

new Product(1, "Laptop", "Electronics"),

new Product(2, "Phone", "Electronics"),

new Product(3, "Shoes", "Fashion"),

new Product(4, "Book", "Stationery"),

new Product(5, "Watch", "Accessories")

};

System.out.println("Linear Search:");

Product linearResult = LinearSearch.search(catalog, "Phone");

System.out.println(linearResult != null ? linearResult : "Product not found");

System.out.println("\nBinary Search:");

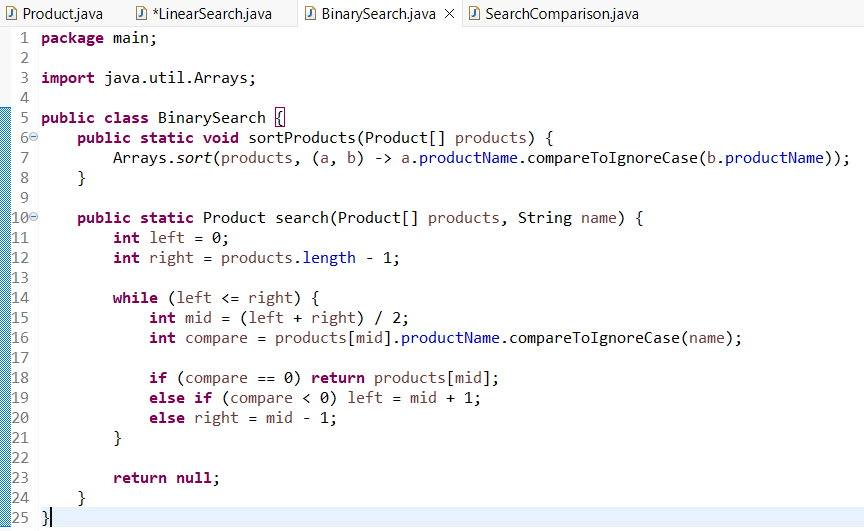
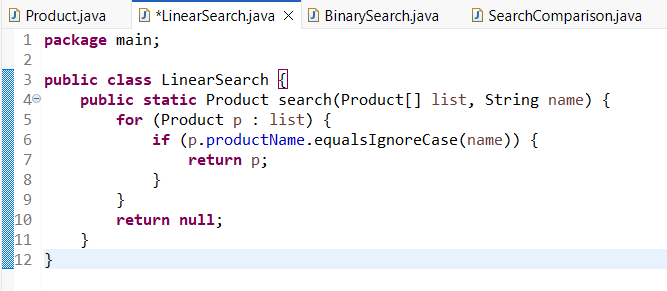
BinarySearch.sortProducts(catalog);

Product binaryResult = BinarySearch.search(catalog, "Phone");

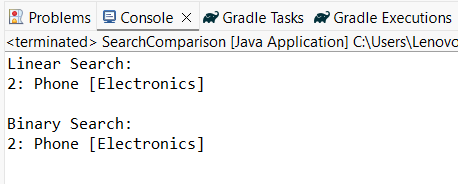
System.out.println(binaryResult != null ? binaryResult : "Product not found");

}

}



**OUTPUT:**



**Exercise 7: Financial Forecasting  
  
Scenario:**

To forecast how much their investment will grow over time based on a fixed interest rate. Implementing a method that multiplies the value each year by (1 + rate) over the given period.  
  
 **CODE:**public class FinanceTool {

public static double calculateFutureValue(double startAmount, double rate, int period) {

double value = startAmount;

for (int i = 1; i <= period; i++) {

value = value \* (1 + rate);

}

return value;

}

public static void main(String[] args) {

double initialAmount = 8000;

double annualRate = 0.07;

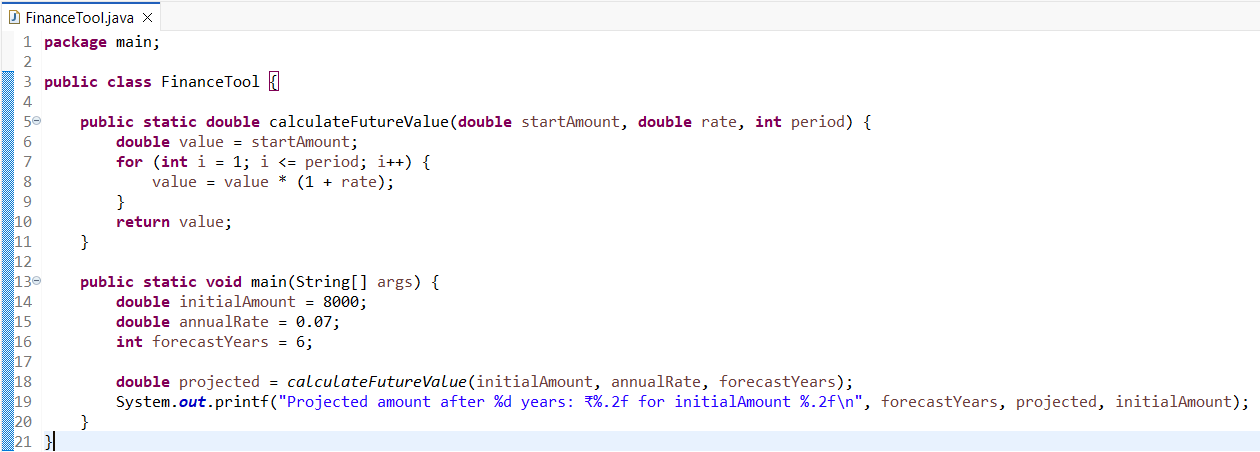
int forecastYears = 6;

double projected = calculateFutureValue(initialAmount, annualRate, forecastYears);

System.out.printf("Projected amount after %d years: ₹%.2f for initialAmount %d\n", forecastYears, projected, initialAmount);

}

}



**OUTPUT:**

