**SUPERSET ID:6412083**

**ALGORITHMS DATA STRUCTURES**

**Exercise 1: E-commerce Platform Search Function**

**CODE:**

package ecommerce;

public class Product {

private int productId;

private String productName;

private String category;

public 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;

}

@Override

public String toString() {

return "[" + productId + "] " + productName + " (" + category + ")";

}

}

package ecommerce;

import java.util.Arrays;

import java.util.Comparator;

public class SearchingAlgorithm {

public static int linearSearch(Product[] products, String targetName) {

for (int i = 0; i < products.length; i++) {

if (products[i].getProductName().equalsIgnoreCase(targetName)) {

return i;

}

}

return -1;

}

public static int binarySearch(Product[] products, String targetName) {

int left = 0, right = products.length - 1;

while (left <= right) {

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

int comparison = products[mid].getProductName().compareToIgnoreCase(targetName);

if (comparison == 0) {

return mid;

} else if (comparison < 0) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return -1;

}

}

package ecommerce;

import java.util.Arrays;

import java.util.Comparator;

public class Main{

public static void main(String[] args) {

Product[] products = {

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

new Product(2, "chocolates", "food"),

new Product(3, "Book", "Stationery"),

new Product(4, "Mobile", "Electronics"),

new Product(5, "Dress", "Fashion")

};

String searchName = "Mobile";

int linearIndex = SearchingAlgorithm.linearSearch(products, searchName);

if (linearIndex != -1) {

System.out.println("Linear Search: Found at index " + linearIndex + " → " + products[linearIndex]);

} else {

System.out.println("Linear Search: Product not found");

}

Arrays.sort(products, Comparator.comparing(p -> p.getProductName().toLowerCase()));

int binaryIndex = SearchingAlgorithm.binarySearch(products, searchName);

if (binaryIndex != -1) {

System.out.println("Binary Search: Found at index " + binaryIndex + " → " + products[binaryIndex]);

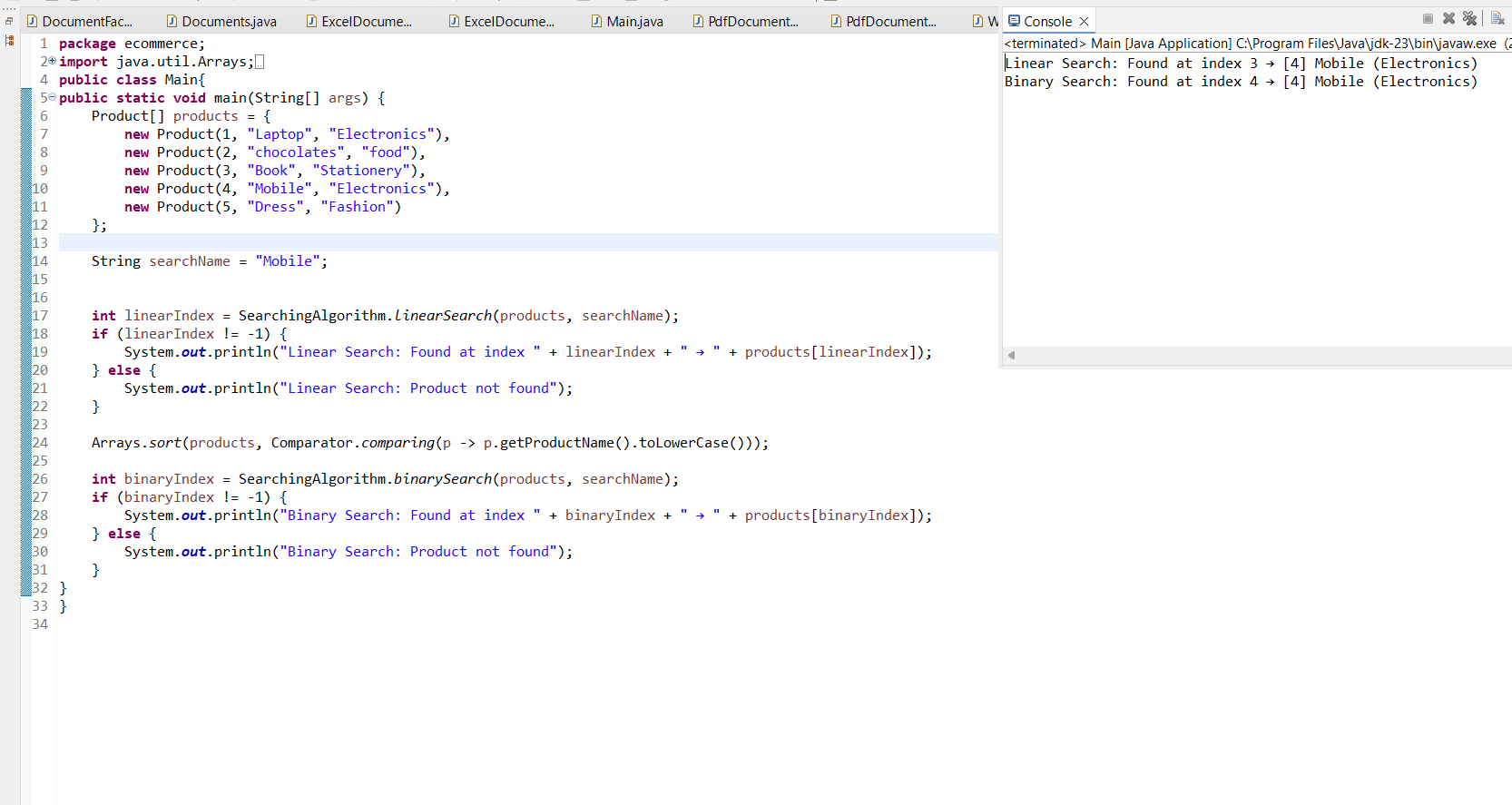
} else {

System.out.println("Binary Search: Product not found");

}

}

}



**Exercise 7: Financial Forecasting**

package financialforecasting;

public class FinancialForecast {

public static double futureValue(double initialAmount, double rate, int years) {

if (years == 0) {

return initialAmount;

}

return *futureValue*(initialAmount, rate, years - 1) \* (1 + rate);

}

public static double futureValueIterative(double initialAmount, double rate, int years) {

double result = initialAmount;

for (int i = 0; i < years; i++) {

result \*= (1 + rate);

}

return result;

}

}

package financialforecasting;

public class Main {

public static void main(String[] args) {

double initialAmount = 1000.0;

double annualGrowthRate = 0.05;

int numberOfYears = 5;

System.*out*.println(" Financial Forecast (Recursive):");

for (int year = 0; year <= numberOfYears; year++) {

double value = FinancialForecast.*futureValue*(initialAmount, annualGrowthRate, year);

System.*out*.printf("Year %d: $%.2f\n", year, value);

}

System.*out*.println(" Optimized Forecast (Iterative):");

for (int year = 0; year <= numberOfYears; year++) {

double value = FinancialForecast.*futureValueIterative*(initialAmount, annualGrowthRate, year);

System.*out*.printf("Year %d: $%.2f\n", year, value);

}

}

}

