FinancialForecasting

ForecastTool.java

public class ForecastTool {

// Recursive method to calculate future value

public static double calculateFutureValue(double presentValue, double growthRate, int years) {

if (years == 0) {

return presentValue;

} else {

return calculateFutureValue(presentValue, growthRate, years - 1) \* (1 + growthRate);

}

}

public static void main(String[] args) {

double presentValue = 10000; // Initial investment

double growthRate = 0.08; // 8% annual growth

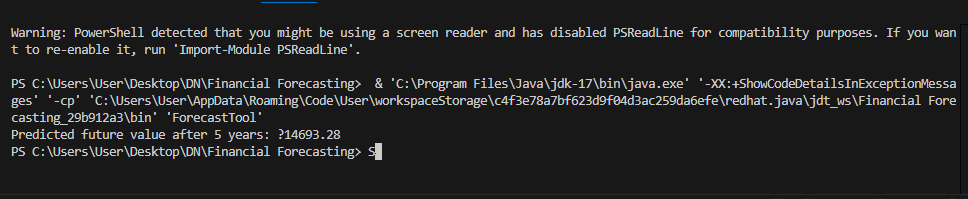
int years = 5; // Forecast for 5 years

double futureValue = calculateFutureValue(presentValue, growthRate, years);

System.out.printf("Predicted future value after %d years: â‚¹%.2f\n", years, futureValue);

}

}



E-Commerce Platform Search Function

SearchFunction.java

import java.util.Arrays;

import java.util.Comparator;

class Product {

int productId;

String productName;

String category;

Product(int id, String name, String cat) {

this.productId = id;

this.productName = name;

this.category = cat;

}

public String toString() {

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

}

}

public class SearchFunction {

public static void main(String[] args) {

Product[] productList = {

new Product(101, "Laptop", "Electronics"),

new Product(205, "Shirt", "Apparel"),

new Product(150, "Shoes", "Footwear"),

new Product(330, "Headphones", "Electronics"),

new Product(120, "Book", "Stationery")

};

int searchId = 150;

// Linear Search

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

Product result1 = linearSearch(productList, searchId);

System.out.println(result1 != null ? result1 : "Product not found");

// Sort by productId for Binary Search

Arrays.sort(productList, Comparator.comparingInt(p -> p.productId));

// Binary Search

System.out.println("\nUsing Binary Search:");

Product result2 = binarySearch(productList, searchId);

System.out.println(result2 != null ? result2 : "Product not found");

}

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

for (Product p : products) {

if (p.productId == id) {

return p;

}

}

return null;

}

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

int low = 0, high = products.length - 1;

while (low <= high) {

int mid = (low + high) / 2;

if (products[mid].productId == id) {

return products[mid];

} else if (products[mid].productId < id) {

low = mid + 1;

} else {

high = mid - 1;

}

}

return null;

}

}

