**Name:** Paarthivi D

**Superset ID:** 6410961

**DN 4.0 - Java FSE Mandatory Hands-on**

**WEEK – 1**

**Algorithms and Data Structures:**

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

**Code:**

package eCommerceSearchFunction;

import java.util.Arrays;

import java.util.Comparator;

public class ECommerceSearch {

// Product class

static 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: " + productId + ", Name: " + productName + ", Category: " + category;

}

}

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;

}

public static void main(String[] args) {

// Sample products

Product[] products = {

new Product(105, "Laptop", "Electronics"),

new Product(101, "Shoes", "Fashion"),

new Product(103, "Smartphone", "Electronics"),

new Product(102, "Watch", "Accessories"),

new Product(104, "Backpack", "Fashion")

};

int searchId = 103;

// Linear Search

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

Product result1 = *linearSearch*(products, searchId);

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

// Sort array before Binary Search

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

// Binary Search

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

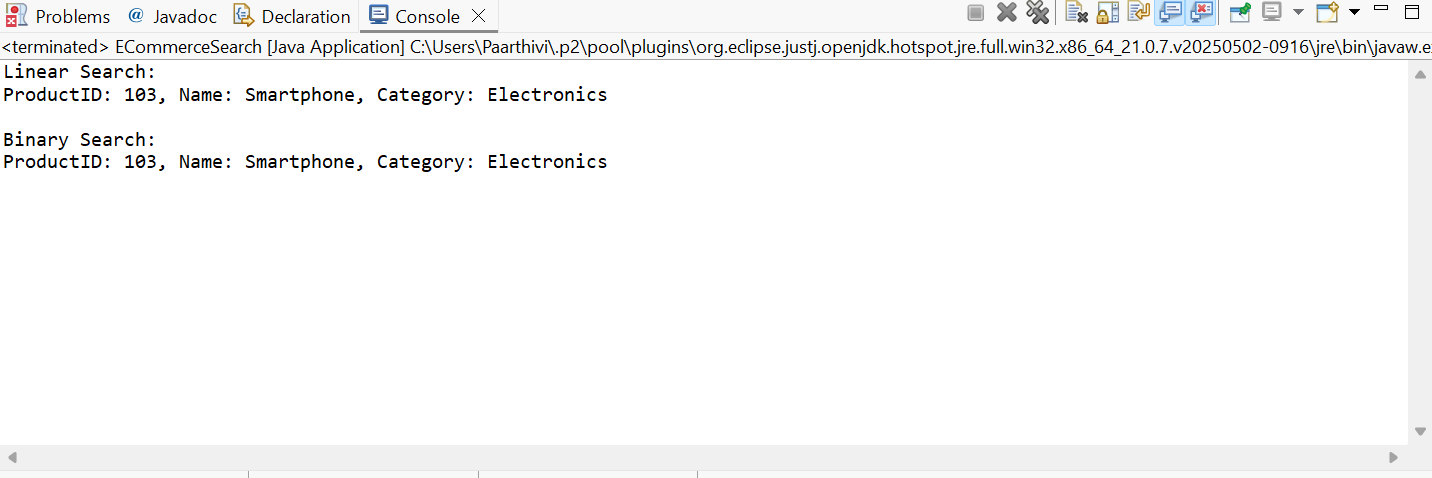
Product result2 = *binarySearch*(products, searchId);

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

}

}

**Output Screenshot:**



**Exercise 7: Financial Forecasting**

**Code:**

package financialForecastingTool;

public class FinancialForecast {

// Recursive Method

public static double calculateFutureValueRecursive(double currentValue, double growthRate, int years) {

if (years == 0) {

return currentValue;

} else {

return (1 + growthRate) \* *calculateFutureValueRecursive*(currentValue, growthRate, years - 1);

}

}

// Iterative Method

public static double calculateFutureValueIterative(double currentValue, double growthRate, int years) {

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

currentValue \*= (1 + growthRate);

}

return currentValue;

}

public static void main(String[] args) {

// Sample input

double currentValue = 10000.0;

double growthRate = 0.08; // 8% annual growth

int years = 5;

// Recursive Calculation

double futureRecursive = *calculateFutureValueRecursive*(currentValue, growthRate, years);

System.*out*.printf("Recursive Prediction: ₹%.2f after %d years%n", futureRecursive, years);

// Iterative Calculation

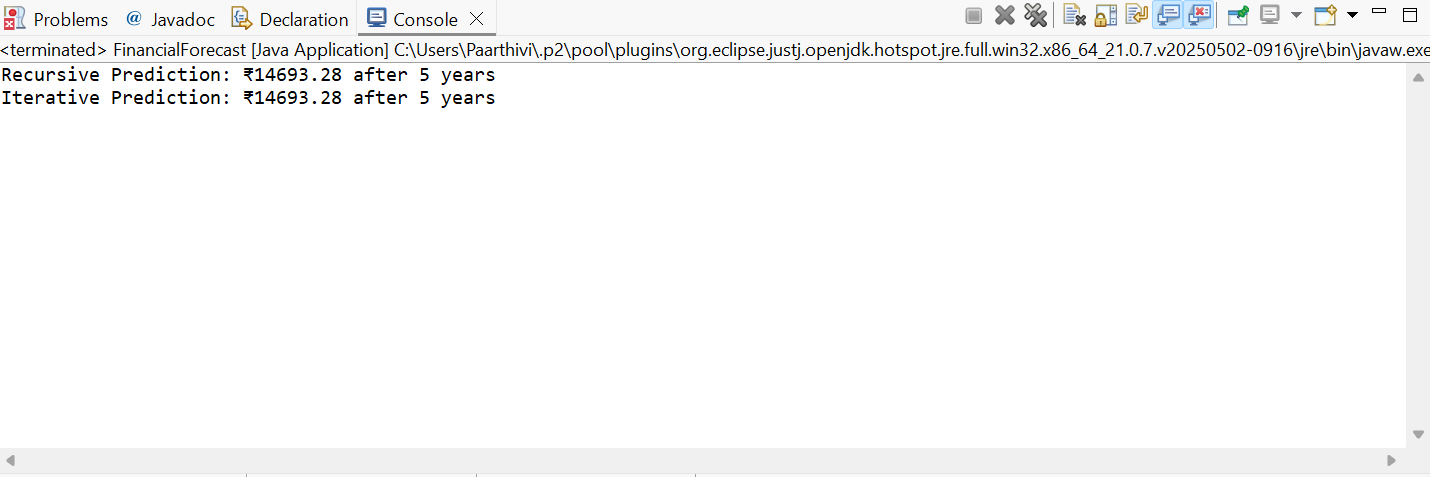
double futureIterative = *calculateFutureValueIterative*(currentValue, growthRate, years);

System.*out*.printf("Iterative Prediction: ₹%.2f after %d years%n", futureIterative, years);

}

}

**Output Screenshot:**

****