Cognizant - DN 4.0 Deep Skilling Java FSE

Week 01 – Algorithms and Data Structures

Superset ID: 6383725

Name: G Ashritha

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

//Product.java

package com.ecommerce.search;

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 + "]";

}

}

//SearchAlgorithms.java

package com.ecommerce.search;

public class SearchAlgorithms {

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

for (Product product : products) {

if (product.getProductName().equalsIgnoreCase(targetName)) {

return product;

}

}

return null;

}

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

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

while (left <= right) {

int mid = (left + right) / 2;

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

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

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

else right = mid - 1;

}

return null;

}

}

//Main.java

package com.ecommerce.search;

import java.util.Arrays;

import java.util.Comparator;

public class Main {

public static void main(String[] args) {

Product[] products = {

new Product(201, "Laptop", "Electronics"),

new Product(202, "Shampoo", "Personal Care"),

new Product(203, "Chair", "Furniture"),

new Product(204, "Phone", "Electronics"),

new Product(205, "Notebook", "Stationery")

};

Product result1 = SearchAlgorithms.linearSearch(products, "Chair");

System.out.println("Linear Search Result: " + (result1 != null ? result1 : "Not Found"));

Arrays.*sort*(products, Comparator.*comparing*(Product::getProductName, String.CASE\_INSENSITIVE\_ORDER));

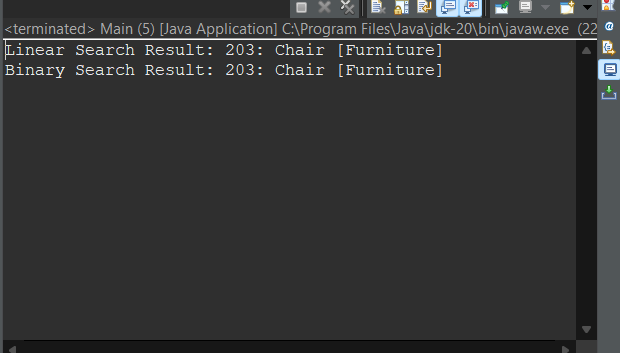
Product result2 = SearchAlgorithms.*binarySearch*(products, "Chair");

System.***out***.println("Binary Search Result: " + (result2 != null ? result2 : "Not Found"));

}

}

//Output



**Exercise 2: Financial Forecasting**

//FinancialForecaster.java

package financial;

public class FinancialForecaster {

public static double forecastRecursive(double presentValue, double rate, int years) {

if (years == 0) {

return presentValue;

}

return (1 + rate) \* *forecastRecursive*(presentValue, rate, years - 1);

}

public static double forecastIterative(double presentValue, double rate, int years) {

double futureValue = presentValue;

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

futureValue \*= (1 + rate);

}

return futureValue;

}

public static void main(String[] args) {

double presentValue = 10000.00;

double annualGrowthRate = 0.08;

int years = 5;

System.out.println("=== Financial Forecasting ===");

System.out.println("Present Value: ₹" + presentValue);

System.out.println("Annual Growth Rate: " + (annualGrowthRate \* 100) + "%");

System.out.println("Time Period: " + years + " years");

double futureRecursive = forecastRecursive(presentValue, annualGrowthRate, years);

System.out.printf("Future Value (Recursive): ₹%.2f\n", futureRecursive);

double futureIterative = forecastIterative(presentValue, annualGrowthRate, years);

System.out.printf("Future Value (Iterative): ₹%.2f\n", futureIterative);

}

}

//Output

