**Week-1 – Engineering Concepts**

**Module - 2 Data Structures & Algorithms**

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

**SearchProduct.java :**

import java.util.\*;

class Product {

// Create a class Product with attributes for searching, such as productId,

// productName, and category.

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 outputString() {

return "Product ID: " + productId + ", Name: " + productName + ", Category: " + category;

}

}

public class SearchProduct {

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

for (Product product : products) {

if (product.productId == productId)

return product;

}

return null;

}

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

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

while (low <= high) {

int mid = low + (high - low) / 2;

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

return products[mid];

else if (products[mid].productId < productId)

low = mid + 1;

else

high = mid - 1;

}

return null;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number of products");

int n = sc.nextInt();

sc.nextLine();

Product[] products = new Product[n];

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

System.out.print("Enter product ID for product " + (i + 1) + ": ");

int productId = sc.nextInt();

sc.nextLine();

System.out.print("Enter product name for product " + (i + 1) + ": ");

String productName = sc.nextLine();

System.out.print("Enter category name for product " + (i + 1) + ": ");

String category = sc.nextLine();

products[i] = new Product(productId, productName, category);

}

Product[] sortedProducts = Arrays.copyOf(products, n);

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

System.out.println("Enter the productID to be searched: ");

int targetId = sc.nextInt();

Product foundLinear = linearSearch(products, targetId);

if (foundLinear != null)

System.out.println("Product found using Linear Search - " + foundLinear.outputString());

else

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

Product foundBinary = binarySearch(sortedProducts, targetId);

if (foundBinary != null)

System.out.println("Product found using Binary Search - " + foundBinary.outputString());

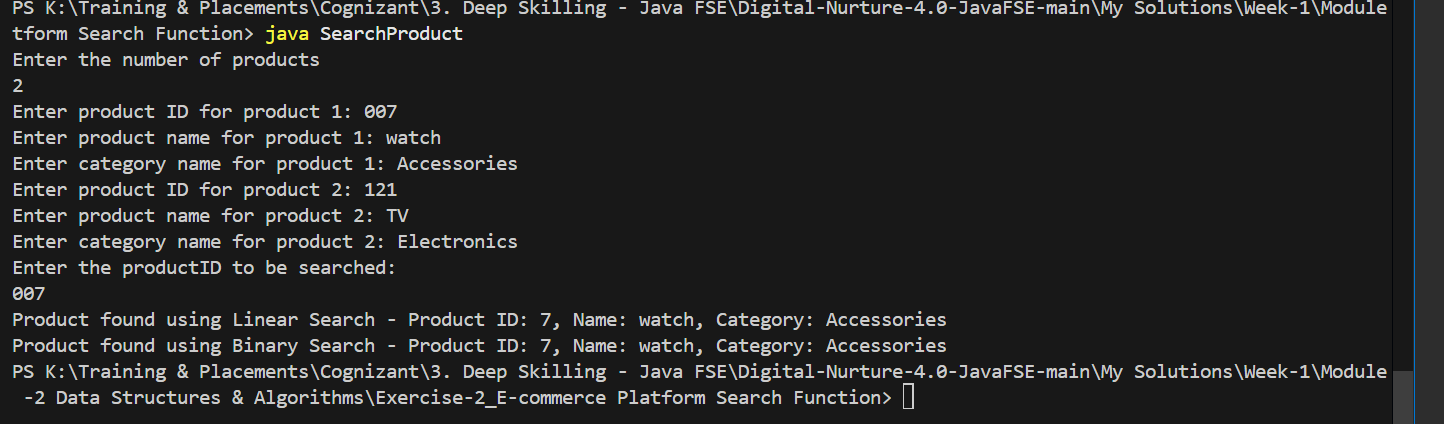
else

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

sc.close();

}

Output -

****

**Exercise 7: Financial Forecasting**

**FinancialForecasting.java :**

import java.util.\*;

public class FinancialForecasting {

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

        if (years == 0)

            return presentValue;

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

    }

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter initial amount: ");

        double presentValue = sc.nextDouble();

        System.out.print("Enter annual growth rate as a decimal: ");

        double growthRate = sc.nextDouble();

        System.out.print("Enter number of years: ");

        int years = sc.nextInt();

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

        System.out.printf("Future Value: %.2f%n", futureValue);

        sc.close();

    }

}

Output:

