**Week 1- Data Structures and Algorithm**

**Risha-6256956**

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

**Product.java**

notepad Product.java

public 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 + " - " + productName + " - " + category + "]";

}

}

**SearchEngine.java**

notepad SearchEngine.java

public 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 + " - " + productName + " - " + category + "]";

}

}

**SearchEngine.java**

notepad SearchEngine.java

import java.util.Arrays;

import java.util.Comparator;

public class SearchEngine {

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

for (Product p : products) {

if (p.productName.equalsIgnoreCase(name)) {

return p;

}

}

return null;

}

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

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

while (low <= high) {

int mid = (low + high) / 2;

int cmp = products[mid].productName.compareToIgnoreCase(name);

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

else if (cmp < 0) low = mid + 1;

else high = mid - 1;

}

return null;

}

public static void sortProducts(Product[] products) {

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

}

}

**Main.java**

notepad Main.java

public class Main {

public static void main(String[] args) {

Product[] products = {

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

new Product(102, "Laptop", "Electronics"),

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

new Product(104, "Watch", "Accessories"),

new Product(105, "Shirt", "Fashion")

};

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

Product result1 = SearchEngine.linearSearch(products, "Phone");

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

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

SearchEngine.sortProducts(products);

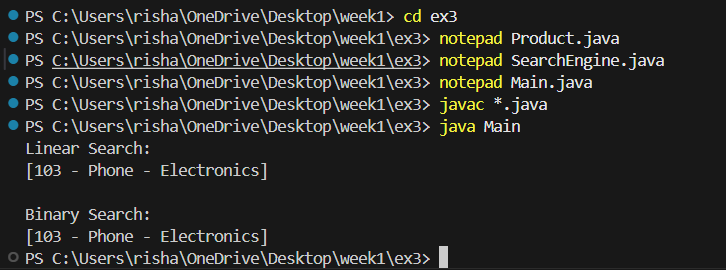
Product result2 = SearchEngine.binarySearch(products, "Phone");

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

}

}

**Output:**



**Exercise 7: Financial Forecasting**

**FinancialForecast.java**

public class FinancialForecast {

    public static double predictFutureValue(int years, double initialAmount, double growthRate) {

        if (years == 0) {

            return initialAmount;

        } else {

            return predictFutureValue(years - 1, initialAmount, growthRate) \* (1 + growthRate);

        }

    }

    public static void main(String[] args) {

        double initialAmount = 10000;

        double annualGrowthRate = 0.08;

        int targetYears = 5;

        double futureValue = predictFutureValue(targetYears, initialAmount, annualGrowthRate);

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

    }

}

