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

**Program:**

**BinarySearchInventory.class**

import java.util.ArrayList;

class BinarySearchInventory {

   ArrayList<Product> products = new ArrayList();

   BinarySearchInventory() {

   }

   void addProduct(Product var1) {

      int var2 = 0;

      int var3 = this.products.size() - 1;

      while(var2 <= var3) {

         int var4 = (var2 + var3) / 2;

         if (((Product)this.products.get(var4)).id < var1.id) {

            var2 = var4 + 1;

         } else {

            var3 = var4 - 1;

         }

      }

      this.products.add(var2, var1);

   }

   boolean searchById(int var1) {

      int var2 = 0;

      int var3 = this.products.size() - 1;

      while(var2 <= var3) {

         int var4 = (var2 + var3) / 2;

         if (((Product)this.products.get(var4)).id == var1) {

            return true;

         }

         if (((Product)this.products.get(var4)).id < var1) {

           var2 = var4 + 1;

         } else {

            var3 = var4 - 1;

         }

      }

      return false;

   }

}

**LinearSearchInventory.class**

**Program:**

import java.util.ArrayList;

import java.util.Iterator;

class LinearSearchInventory {

   ArrayList<Product> products = new ArrayList();

   LinearSearchInventory() {

   }

   void addProduct(Product var1) {

      this.products.add(var1);

   }

   boolean searchById(int var1) {

      Iterator var2 = this.products.iterator();

      Product var3;

      do {

         if (!var2.hasNext()) {

            return false;

         }

         var3 = (Product)var2.next();

      } while(var3.id != var1);

      return true;

   }

}

**E\_Commerce.java**

**Program:**

import java.util.\*;

class Product {

    int id;

    String name;

    int quantity;

    double price;

    Product(int id, String name, int quantity, double price) {

        this.id = id;

        this.name = name.toLowerCase();

        this.quantity = quantity;

        this.price = price;

    }

}

class LinearSearchInventory {

    ArrayList<Product> products = new ArrayList<>();

    void addProduct(Product product) {

        products.add(product);

    }

    boolean searchById(int id) {

        for (Product product : products) {

            if (product.id == id) return true;

        }

        return false;

    }

}

class BinarySearchInventory {

    ArrayList<Product> products = new ArrayList<>();

    void addProduct(Product product) {

        int low = 0, high = products.size() - 1;

        while (low <= high) {

            int mid = (low + high) / 2;

            if (products.get(mid).id < product.id) {

                low = mid + 1;

            } else {

                high = mid - 1;

            }

        }

        products.add(low, product);

        }

    boolean searchById(int id) {

        int low = 0, high = products.size() - 1;

        while (low <= high) {

            int mid = (low + high) / 2;

            if (products.get(mid).id == id) return true;

            else if (products.get(mid).id < id) low = mid + 1;

            else high = mid - 1;

        }

        return false;

    }

}

public class E\_Commerce {

    public static void main(String[] args) {

        Product mobile = new Product(1, "mobile", 10, 20.00);

        Product batteries = new Product(2, "batteries", 20, 1.50);

        Product chargers = new Product(3, "chargers", 30, 2.00);

        LinearSearchInventory linearInventory = new LinearSearchInventory();

        BinarySearchInventory binaryInventory = new BinarySearchInventory();

        linearInventory.addProduct(mobile);

        linearInventory.addProduct(batteries);

        linearInventory.addProduct(chargers);

        binaryInventory.addProduct(mobile);

        binaryInventory.addProduct(batteries);

        binaryInventory.addProduct(chargers);

        if (linearInventory.searchById(2)) {

            System.out.println("Product with ID 2 found using Linear Search.");

        } else {

            System.out.println("Product with ID 2 not found using Linear Search.");

        }

        if (binaryInventory.searchById(2)) {

            System.out.println("Product with ID 2 found using Binary Search.");

        } else {

            System.out.println("Product with ID 2 not found using Binary Search.");

**}**

    }

}



**Exercise 7: Financial Forecasting**

**Program:**

public class FutureValueCalculator {

    public static void main(String[] args) {

        double initialInvestment = 15000;

        double[] annualGrowthRates = {0.05, 0.04, 0.09, 0.06, 0.03, 0.02, 0.01};

        double futureValue = calculateFutureValue(initialInvestment, annualGrowthRates, 0);

        System.out.printf("Future value after %d years: %.2f\n", annualGrowthRates.length, futureValue);

    }

    public static double calculateFutureValue(double currentValue, double[] growthRates, int yearIndex) {

        if (yearIndex == growthRates.length) {

            return currentValue;

        }

        double updatedValue = currentValue \* (1 + growthRates[yearIndex]);

        return calculateFutureValue(updatedValue, growthRates, yearIndex + 1);

    }

}

****