**Data Structures and Algorithms**

**1.E-commerce Platform Search Function**

**Product.java**

package InventoryManagementSystem;

public class Product {

int productId;

String productname;

String category;

Product(int productId,String productname,String category){

this.productId=productId;

this.productname=productname;

this.category=category;

}

}

**Linearsearch.java**

package InventoryManagementSystem;

import java.util.\*;

public class linearsearch {

public long linearsearching(Product[] product,String choose) {

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

long start\_time=System.*nanoTime*();

int flag=0;

for(Product i:product) {

if(i.productname.equalsIgnoreCase(choose)) {

System.*out*.println("Product Found");

flag=1;

}

}

if(flag==0) {

System.*out*.println("Product not Found");

}

long end\_time=System.*nanoTime*();

long duration=end\_time-start\_time;

return duration;

}

}

**Binarysearch.java**

package InventoryManagementSystem;

import java.util.\*;

public class binarysearch {

public long binarysearching(Product[] product,String choose) {

System.*out*.println("binary Search");

Arrays.*sort*(product, new Comparator<Product>() {

public int compare(Product p1, Product p2) {

return p1.productname.compareToIgnoreCase(p2.productname);

}

})

long start\_time=System.*nanoTime*();

int flag=0;

int left=0;

int right=product.length-1;

while(left<=right) {

int mid=(left+right)/2;

String name=product[mid].productname;

int num=name.compareToIgnoreCase(choose);

if(num==0) {

System.*out*.println("Product found");

flag=1;

break;

}

else if(num<0) {

left=mid+1;;

}

else {

right=mid-1;

}

}

if(flag==0) {

System.*out*.println("Product not found");

}

long end\_time=System.*nanoTime*();

long duration=end\_time-start\_time;

return duration;

}

}

**Mainp.java**

package InventoryManagementSystem;

import java.util.\*;

public class mainp {

public static void main(String[] args) {

Scanner sc=new Scanner(System.*in*);

Product[] product= {

new Product(1,"SonyTv","Electronics"),

new Product(2,"MiTv","Electronics"),

new Product(3,"car","vehicle"),

new Product(4,"bus","vehicle")};

System.*out*.println("Enter the product name to search the product");

String choose=sc.nextLine();

System.*out*.println();

linearsearch l=new linearsearch();

long linear\_time=l.linearsearching(product,choose);

System.*out*.println();

binarysearch b=new binarysearch();

long binary\_time=b.binarysearching(product,choose);

System.*out*.println();

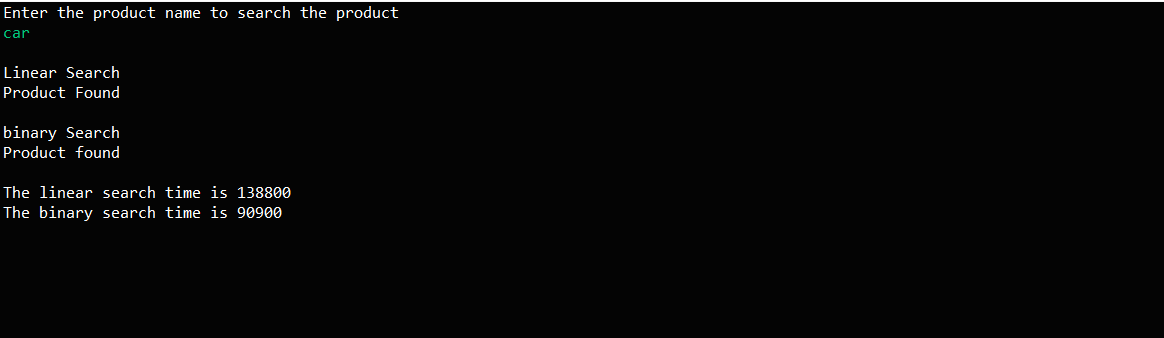
System.*out*.println("The linear search time is "+ linear\_time);

System.*out*.println("The binary search time is "+ binary\_time);

}

}

**Output**

****

**2. Financial Forecasting**

**CalculateFutureValue.java**

package FinancialForecasting;

import java.util.\*;

public class CalculateFutureValue {

public static double calculateAverageGrowthRate(double[] pastProfits) {

double totalGrowth = 0;

for (int i = 1; i < pastProfits.length; i++) {

double growth = (pastProfits[i] - pastProfits[i - 1]) / pastProfits[i - 1];

totalGrowth += growth;

}

return totalGrowth / (pastProfits.length - 1);

}

public static double forecastFutureProfit(int year, double lastProfit, double rate) {

if (year == 0) return lastProfit;

return forecastFutureProfit(year - 1, lastProfit, rate) \* (1 + rate);

}

public static void main(String[] args) {

double[] pastProfits = {10000, 12000, 15000, 18000};

double lastKnownProfit = pastProfits[pastProfits.length - 1];

double avgGrowthRate = calculateAverageGrowthRate(pastProfits);

System.out.printf("Average Growth Rate: %.2f%%\n\n", avgGrowthRate \* 100);

int forecastYears = 5;

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

double predictedProfit = forecastFutureProfit(i, lastKnownProfit, avgGrowthRate);

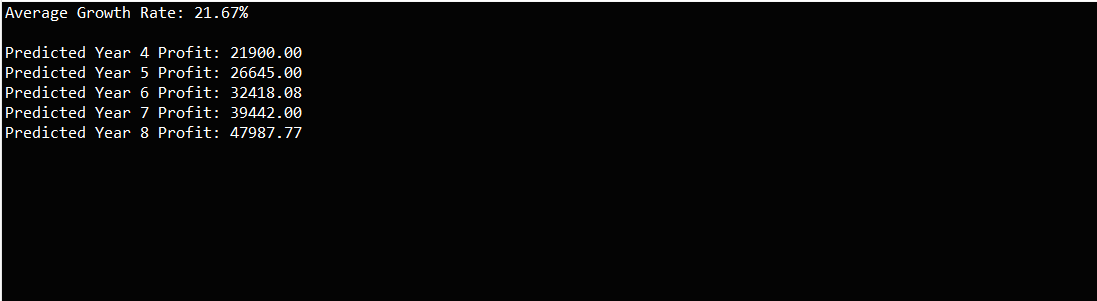
System.out.printf("Predicted Year %d Profit: %.2f\n", i + pastProfits.length - 1, predictedProfit);

}

}

}

**Output**

****