**Algorithms Data Structures**

**E-commerce Platform Search Function**

**Code:**

using System;

using System.Linq;

class Program

{

    static void Main()

    {

        Product[] products = {

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

            new Product(2, "Shirt", "Clothing"),

            new Product(3, "Phone", "Electronics"),

            new Product(4, "Book", "Education")

        };

        Console.WriteLine("Linear Search:");

        var result1 = LinearSearch(products, "Phone");

        Console.WriteLine(result1 != null ? $"Found: {result1.ProductName}" : "Not Found");

        var sortedProducts = products.OrderBy(p => p.ProductName).ToArray();

        Console.WriteLine("Binary Search:");

        var result2 = BinarySearch(sortedProducts, "Phone");

        Console.WriteLine(result2 != null ? $"Found: {result2.ProductName}" : "Not Found");

    }

    static Product LinearSearch(Product[] products, string name)

    {

        foreach (var product in products)

        {

            if (product.ProductName.Equals(name, StringComparison.OrdinalIgnoreCase))

                return product;

        }

        return null;

    }

    static Product BinarySearch(Product[] products, string name)

    {

        int left = 0;

        int right = products.Length - 1;

        while (left <= right)

        {

            int mid = (left + right) / 2;

            int cmp = string.Compare(products[mid].ProductName, name, StringComparison.OrdinalIgnoreCase);

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

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

            else right = mid - 1;

        }

        return null;

    }

}

public class Product

{

    public int ProductId { get; set; }

    public string ProductName { get; set; }

    public string Category { get; set; }

    public Product(int id, string name, string category)

    {

        ProductId = id;

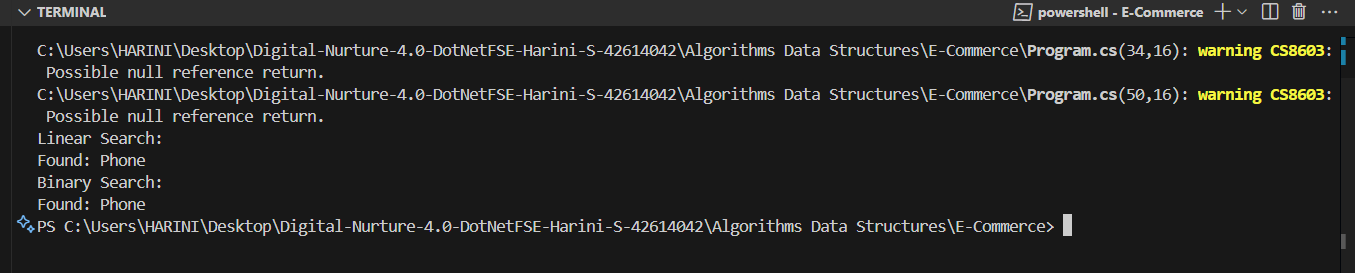
        ProductName = name;

        Category = category;

    }

}

**Output:**



**Financial Forecasting**

**Code:**

using System;

class Program

{

    static void Main()

    {

        double presentValue = 1000.0;

        double annualGrowthRate = 0.05;

        int years = 5;

        double futureValue = ForecastValue(presentValue, annualGrowthRate, years);

        Console.WriteLine($"Forecasted value after {years} years: {futureValue:F2}");

    }

    static double ForecastValue(double currentValue, double rate, int years)

    {

        if (years == 0)

            return currentValue;

        return ForecastValue(currentValue \* (1 + rate), rate, years - 1);

    }

}

**Output:**

