**WEEK 1   
  
 ALGORITHMS AND DATA STRUCTURES  
  
 EXERCISE 2 - E-COMMERCE PLATFORM   
 SEARCH FUNCTION**  
  
  
**STEP 1 - Product.cs :**

namespace ECommerceSearchExample

{

    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;

        }

    }

}

**STEP 2 - Program.cs :**

using System;

using System.Linq;

namespace ECommerceSearchExample

{

    class Program

    {

        static void Main(string[] args)

        {

            Product[] products = new Product[]

            {

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

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

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

                new Product(4, "Shoes", "Footwear"),

                new Product(5, "Watch", "Accessories")

            };

            string target = "Phone";

            Product linearResult = SearchEngine.LinearSearch(products, target);

            Console.WriteLine(linearResult != null ? $"Linear: {linearResult.ProductName} found." : "Linear: Not found.");

            Product[] sortedProducts = products.OrderBy(p => p.ProductName).ToArray();

            Product binaryResult = SearchEngine.BinarySearch(sortedProducts, target);

            Console.WriteLine(binaryResult != null ? $"Binary: {binaryResult.ProductName} found." : "Binary: Not found.");

        }

    }

}

**STEP 3 - SearchEngine.cs :**

using System;

using System.Collections.Generic;

namespace ECommerceSearchExample

{

    public class SearchEngine

    {

        public static Product LinearSearch(Product[] products, string target)

        {

            foreach (var product in products)

            {

                if (product.ProductName == target)

                {

                    return product;

                }

            }

            return null;

        }

        public static Product BinarySearch(Product[] products, string target)

        {

            int left = 0;

            int right = products.Length - 1;

            while (left <= right)

            {

                int mid = (left + right) / 2;

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

                if (compare == 0)

                    return products[mid];

                else if (compare < 0)

                    left = mid + 1;

                else

                    right = mid - 1;

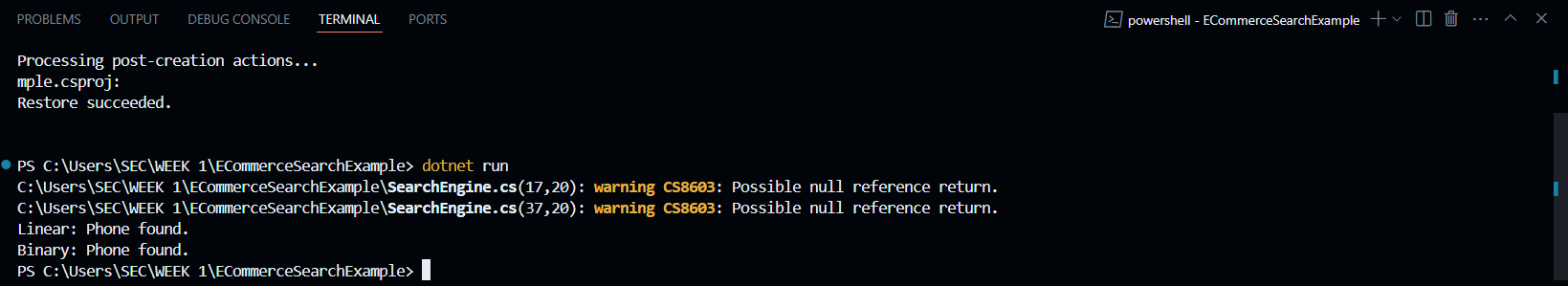
            }

            return null;

        }

    }

}

**OUTPUT :   
  
**