**WEEK – 1 Assignments**

**Data structures and Algorithms**

**Mandatory hands-on :-**

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

**Scenario:**

You are working on the search functionality of an e-commerce platform. The search needs to be optimized for fast performance.

**Answers :**

Big O notation :-

Big O notation describes algorithm efficiency by measuring how runtime grows relative to input size (n).

* Best-case: Minimum operations needed
* Average-case: Expected performance with random inputs
* Worst-case: Maximum operations needed

For search operations:

* Linear search:  
  O(1) best-case, O(n) average/worst-case
* Binary search:  
  O(1) best-case, O(log n) average/worst-case

## **Product Class Implementation :-**

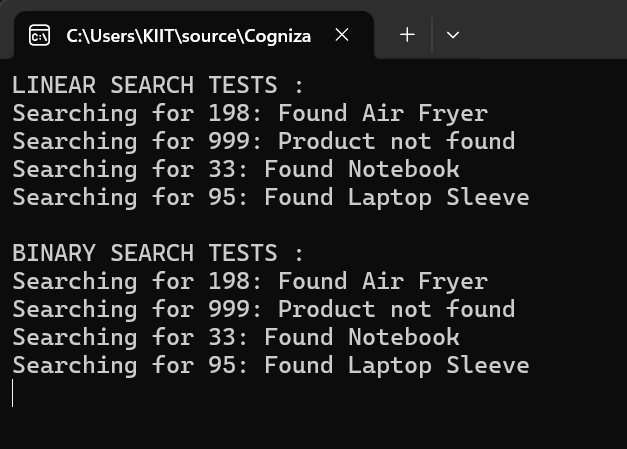
**Codes :-**

**File names :**

**Program.cs :**

| using System;  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;  } }  public class ProductSearch {  // Linear search  public static Product LinearSearch(Product[] products, int targetId)  {  foreach (var product in products)  {  if (product.ProductId == targetId)  return product;  }  return null;  }   // Binary search  public static Product BinarySearch(Product[] sortedProducts, int targetId)  {  int left = 0;  int right = sortedProducts.Length - 1;   while (left <= right)  {  int mid = left + (right - left) / 2;  if (sortedProducts[mid].ProductId == targetId)  return sortedProducts[mid];  if (sortedProducts[mid].ProductId < targetId)  left = mid + 1;  else  right = mid - 1;  }  return null;  } }  class Program {  static void Main()  {  Product[] allProducts = {  new Product(102, "Wireless Mouse", "Electronics"),  new Product(205, "Coffee Maker", "Kitchen"),  new Product(87, "Running Shoes", "Sports"),  new Product(301, "Desk Lamp", "Home"),  new Product(45, "Water Bottle", "Sports"),  new Product(156, "Bluetooth Speaker", "Electronics"),  new Product(78, "Yoga Mat", "Fitness"),  new Product(210, "Toaster", "Kitchen"),  new Product(33, "Notebook", "Stationery"),  new Product(189, "Backpack", "Travel"),  new Product(267, "Digital Camera", "Electronics"),  new Product(54, "Sunglasses", "Fashion"),  new Product(198, "Air Fryer", "Kitchen"),  new Product(76, "Dumbbell Set", "Fitness"),  new Product(123, "External SSD", "Computers"),  new Product(289, "Gaming Chair", "Furniture"),  new Product(61, "Wireless Earbuds", "Audio"),  new Product(142, "Electric Toothbrush", "Health"),  new Product(224, "Smart Watch", "Wearables"),  new Product(95, "Laptop Sleeve", "Accessories")  };   Product[] sortedProducts = new Product[allProducts.Length];  Array.Copy(allProducts, sortedProducts, allProducts.Length);  Array.Sort(sortedProducts, (p1, p2) => p1.ProductId.CompareTo(p2.ProductId));   // Test  Console.WriteLine("LINEAR SEARCH TESTS :");  TestSearch(allProducts, 198, "Linear");   TestSearch(allProducts, 999, "Linear"); // Non-existing item  TestSearch(allProducts, 33, "Linear");   TestSearch(allProducts, 95, "Linear");    Console.WriteLine("\nBINARY SEARCH TESTS :");  TestSearch(sortedProducts, 198, "Binary");   TestSearch(sortedProducts, 999, "Binary"); // Non-existing item  TestSearch(sortedProducts, 33, "Binary");   TestSearch(sortedProducts, 95, "Binary");   Console.ReadLine();  }   static void TestSearch(Product[] products, int targetId, string method)  {  Product result = method == "Linear" ?  ProductSearch.LinearSearch(products, targetId) :  ProductSearch.BinarySearch(products, targetId);   Console.WriteLine($"Searching for {targetId}: " +  (result != null ? $"Found {result.ProductName}" : "Product not found"));  } } |
| --- |

**Output :-**

****

**Which is best ?**

Binary search is more suitable for an e-commerce platform because it is much faster (O(log n) time) than linear search (O(n) time) when searching large, sorted product catalogs. E-commerce databases are usually sorted or indexed, making binary search the best choice for quick and efficient product lookups. Linear search is only practical for very small or unsorted datasets.