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

## ****Step 1: Understand Asymptotic Notation****

**Big O Notation** gives the upper bound of an algorithm's time or space complexity as input grows.

| **Search Type** | **Best Case** | **Average Case** | **Worst Case** |
| --- | --- | --- | --- |
| Linear Search | O(1) | O(N) | O(N) |
| Binary Search | O(1) | O(log N) | O(log N) |

**Linear Search:** Works on any array (unsorted). Simple but slow for large data.

**Binary Search:** Needs sorted data. Very fast even for large datasets.

**Step 2: Setup Product Class**

public class Product

{

public int ProductId { get; set; }

public string ProductName { get; set; }

public string Category { get; set; }

public Product(int productId, string productName, string category)

{

ProductId = productId;

ProductName = productName;

Category = category;

}

}

**Step 3:Implement Search Algorithms**

**Linear search:**

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

{

foreach (var product in products)

{

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

{

return product;

}

}

return null;

}

**Step 4:Binary Search:**

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

{

int left = 0;

int right = products.Length - 1;

while (left <= right)

{

int mid = (left + right) / 2;

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

if (comparison == 0)

return products[mid];

else if (comparison < 0)

left = mid + 1;

else

right = mid - 1;

}

return null;

}

Testing Everything:

using System;

class Program

{

static void Main()

{

Product[] products = new Product[]

{

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

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

new Product(3, "Shoes", "Fashion"),

new Product(4, "Watch", "Accessories"),

new Product(5, "Bag", "Fashion"),

};

// Sort products by name for binary search

Array.Sort(products, (p1, p2) => p1.ProductName.CompareTo(p2.ProductName));

Console.WriteLine("Enter product name to search:");

string searchName = Console.ReadLine();

// Linear Search

Product result1 = LinearSearch(products, searchName);

if (result1 != null)

Console.WriteLine($"Linear Search: Found {result1.ProductName} in {result1.Category}");

else

Console.WriteLine("Linear Search: Product not found");

// Binary Search

Product result2 = BinarySearch(products, searchName);

if (result2 != null)

Console.WriteLine($"Binary Search: Found {result2.ProductName} in {result2.Category}");

else

Console.WriteLine("Binary Search: Product not found");

}

}

## ****Step 5: Analysis****

| **Algorithm** | **Time Complexity** | **Space Complexity** |
| --- | --- | --- |
| Linear Search | O(N) | O(1) |
| Binary Search | O(log N) | O(1) |

**Linear Search:** Suitable if data is unsorted or data changes frequently.

**Binary Search:** Best for large, sorted datasets where data doesn't change frequently.