using System;

using System.Collections.Generic;

using System.Linq;

namespace MergedList

{

class Program

{

/\*

\* Given two lists of students, return a merged list of the two lists.

\* Order of the merged list is not important.

\* Example:

\* List A = [Allison, Brian, Peter]

\* List B = [Jason, Peter, Sara]

\* Return = [Allison, Brian, Peter, Jason, Sara]

\*/

static void Main(string[] args)

{

List<string> arr1 = new List<string>(){ "Allison", "Brian", "Peter" };

List<string> arr2 = new List<string> { "Jason", "Peter", "Sara" };

List<string> result = MergedListNoDuplicates(arr1, arr2);

foreach(var item in result)

Console.Write($"{item} ");

}

/\*

\* Approach:

\* 1. Copy the first list to the result list

\* 2. Loop through two lists for each, then compare the component.

\* 3. If the two elements are duplicated, remove the element from the copied list.

\* 4. Print out the result.

\*/

public static List<string> MergedListNoDuplicates(List<string> a, List<string> b)

{

List<string> result = new List<string>(a); // copy the list

for (int i = 0; i < b.Count; i++) // Time Complexity = O(n)

{

for (int j = 0; j < result.Count; j++) // Time Complexity = O(n^2)

{

if (b[i] == result[j]) // Time Complexity = O(n^2 + m)

result.Remove(result[j]);

}

result.Add(b[i]);

}

return result;

/\*

\* Another Approach:

\* Use Hashset to display only unique components (no duplicates).

\* Can improve time complexitiy to O(1) from O(n^2 + m), constant time.

\*/

//HashSet<string> result = new HashSet<string>(a);

//result.UnionWith(b);

//return result.ToList();

}

}

}