Common Items

Description

Let *A* and *B* be two sets of integers, represented as unsorted arrays. Suppose that $|A| = n$, $|B| = m$, and $n \leq m$. Write an efficient algorithm to find ALL common elements between *A* and *B (if* any).

Complexity

The complexity of your algorithm should be **less than O(mn).**

Function: **Implement it!**

```
static public int [] RequiredFuntion(int[] arr1, int[] arr2)
```

`PROBLEM_CLASS.cs` includes this method.

RETURN array of common element (if any) or empty array if no common elements.
To return an empty array: `return new int[] { }`

Example

Input:

Array1 = [1, 8, -1]

Array2 = [1, 9, -1, 15, 18, 33, 0, 4, 7]

Output: [1, -1]

Input:

Array1 = [1]

Array2 = [-1]

Output: []

Input:

Array1 = [6, 7, 8, 9, 10]

Array2 = [-1, 0, 1, 2, 3, 4, 5, 6]

Output: [6]

Input:

Array1 = [1, 3, 5, 9, 11, 13, 15, 17, 19]

Array2 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Output: [1, 3, 5, 9]

C# Help

Getting the size of 1D array

```
int size = array1D.GetLength(0);
```

Getting the size of 2D array

```
int size1 = array2D.GetLength(0);

int size2 = array2D.GetLength(1);
```

Creating 1D array

```
int [] array1D = new int [size]
```

Creating 2D array

```
int [,] array2D = new int [size1, size2]
```

Sorting single array

Sort the given array "items" in ascending order

```
Array.Sort(items);
```

Sorting parallel arrays

Sort the first array "master" and re-order the 2nd array "slave" according to this sorting

```
Array.Sort(master, slave);
```