Modulo Sum

Description

Given a sequence of positive integers $a_1, a_2, \ldots a_n$, and a positive integer M. Your goal is to determine if a non-empty subsequence of $a_1, a_2, \ldots a_n$ that the sum of numbers in this subsequence is divisible by m.

Requirements:

Implement the following TWO functions in **MOST EFFICIENT WAY** (Hint: make sure to select the **suitable solution direction**),

1. 1st function: return whether there's a subsequence that is divisible by M or not.
2. 2nd function: return the subsequence itself (if found), or null (if no such one).

Function:

First Function:

```
bool SolveValue(int[] items, int N, int M)
<returns>true if there's subsequence with sum divisible by M... false otherwise
```

Second Function:

```
int[] ConstructSolution(int[] items, int N, int M)
<returns>if exists, return the numbers themselves whose sum is divisible by 'M'
else, return null
```

Example

#	Input	Output	Subsequence
1	M = 5, items = [1,2,3]	true	2, 3
2	M = 6, items = [5]	false	Null
3	M = 6, items = [3,1,1,3]	true	3, 3
4	M = 6, items = [6,6,6,6,6,6]	true	6

C# Help

Dictionary (Hash)

Creation

To create a dictionary of a certain key (e.g. string) and value (e.g. array of strings)

```
//default initial size
```

```
Dictionary<string, string[]> myDict1 = new  Dictionary<string, string[]>();

//given initial size
Dictionary<string, string[]> myDict2 = new  Dictionary<string, string[]>(size);
```

Manipulation

1. `myDict1.Count` ➔ Get actual number of items in the dictionary
2. `myDict1[key]` ➔ Get/Set the value associated with the given key in the dictionary
3. `myDict1.Add(key, value)` ➔ Add the specified key and value to the dictionary
4. `myDict1.Remove(key)` ➔ Remove the value with the specified key from the dictionary
5. `myDict1.ContainsKey(key)` ➔ Check if the specified key exists in the dictionary

ARRAYS:

Creating 1D array

```
int [] array = new int [size]
```

Creating 2D array

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

Length of 1D array

```
int arrayLength = my1DArray.Length
```

Length of 2D array

```
int array1stDim = my2DArray.GetLength(0)

int array2ndDim = my2DArray.GetLength(1)
```

Sorting single array

Sort the given array 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);
```