Text Plagiarism

Description

Given a paragraph and a complete text, it's required to calculate the plagiarism similarity of the given paragraph vs the given text. The Plagiarism similarity is defined as the max common subsequence of words between the given paragraph and EACH paragraph in the given text. Comparison is case IN-SENSITIVE (i.e. Cat = CAT = cat = CaT)

Definitions:

1. **Word:** a set of continuous characters separated by space or tab (Words separator: ' ' or '\t')
2. **Paragraph** in Text: any continuous set of words/chars ended by new line(s) (Paragraphs separator: '\n' or '\r')
3. **A subsequence** of a given sequence is just the given sequence with some elements (possibly none) left out.

Requirements:

Implement TWO functions,

1. First function: return the value of the plagiarism similarity.
2. Second function: return the subsequence (if any) or null.

Function:

First Function:

```
                int SolveValue(string paragraph, string text)
<returns>Plagiarism similarity between the query paragraph and the complete text
```

Second Function:

```
            string[] ConstructSolution(string paragraph, string text)
<returns>the common subsequence words themselves (if any) or null if no common
words
```

Example

```
paragraph = hello world how are you
text = hello are you world how
Plagiarism Similarity = 3
Subsequence = hello are you
```

paragraph = DP is a careful brute force and a complete D&C with overlapped sub-problems
text =
Greedy has two conditions: optimal substructure and safe greedy choice
DP has two conditions: optimal substructure (i.e. D&C) and overlapped subproblems

Plagiarism Similarity = 3
Subsequence = DP and overlapped (matched with 2nd paragraph)

paragraph = Algorithms Analysis and Design
text =
ANALYSIS & DESIGN of ALGORITHMS
System analysis and design
Numerical Analysis
Data Structures
S/W DESIGN
Plagiarism Similarity = 3
Subsequence = analysis and design (matched with 2nd paragraph)

C# Help

STRINGS:

Creation

```
1. string str = string.Empty //create empty string
2. string str = "[string characters]" //create and initialize string
```

Manipulation

1. `str.Length` ➔ get the number of chars in the string
2. `str.Split(separators, StringSplitOptions.RemoveEmptyEntries)`

   ➔ Split the string according to the given character separators and remove any empty string from the results.
3. `str.ToLower()` or `str.ToUpper()` ➔ convert all chars to lower/upper case
4. `Str1 == Str2` ➔ compare the two strings for equality

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);
```