**C# .Net Coding Test**

Public Git Repo:

**Question 1 - Overlapping Appointments**

**Project: ApptOverlap**

*\*\*\* Add your code to the “isOverlapping” method in Program.cs*

*where it says ///INSERT YOUR CODE HERE*

**Programming Challenge Description**

Implement a function that will return a Boolean indicating whether or not two appointments are overlapping. If one appointment ends at the same time that the other starts, consider them to not be overlapping (since typically you wouldn’t consider an appointment that ends at 5 p.m. to be conflicting with an appointment that starts at 5 p.m.).

**Input**

Four integers range 0 - 23 (inclusive), \*\*\_appointment1Start, appointment1End, appointment2Start, appointment2End\_\*\*, denoting the starting and end hours of the appointments.

You may assume that \*\*\_appointment1Start\_\*\* is before or equal to \*\*\_appointment1End\_\*\*, and \*\*\_appointment2Start\_\*\* is before or equal to \*\*\_appointment2End\_\*\*.

**Output**

A boolean indicating if the two appointments are overlapping.

**Expected Output**

Test 1

Input: 1 2 2 4

Output: 0

Test 2

Input: 1 3 2 4

Output: 1

**Question 2 - Find a sequence**

**Project: NeedleHaystack**

*\*\*\* Add your code to the “findSequence” method in Program.cs*

*where it says ///INSERT YOUR CODE HERE*

**Programming Challenge Description**

Implement a function that takes two arrays of strings as input.  The function should search the first array, which we’ll call the \_haystack\_, for the sequence contained in the second array, which we’ll call the \_needle\_.  If the sequence in the \_needle\_ is found within the \_haystack\_, then the function should return the index of the position of the sequence in the \_haystack\_.  If the sequence in the \_needle\_ is not found, then the function should return \_\-1\_.

For example, if the haystack were {“one”, “two”, “three”, “four”} and the needle were {“two”, “three”}, then the function should return 1, since the sequence in the needle {“two”, “three”} is positioned at the second position of the haystack (assume that the arrays are 0-indexed).

**Input**

Two string arrays, \_haystack \_and \_needle\_\_.\_

**Output**

A single integer which represents the starting index of the \_needle\_ in the \_haystack\_, or \_\-1\_ if the \_needle\_ is not found.

**Expected Output**

Test 1

Input:

3

apple

banana

pear

2

banana

pear

Output: 1

Test 2

Input:

3

apple

banana

pear

2

four

five

Output: -1

**Question 3 - Fun with Anagrams**

**Project: NeedleHaystack**

*\*\*\* Add your code to the “funWithAnagrams” method in Program.cs*

*where it says ///INSERT YOUR CODE HERE*

**Programming Challenge Description**

Implement a function \_funWithAnagrams\_ that will take an array of strings as input. The function should remove each string that is an anagram of an earlier string, then return the remaining array in sorted order. Two strings are anagrams if they are permutations of each other.

For example, given the strings \_s = ['code', 'doce', 'ecod', 'framer', 'frame'],\_  the strings \_'doce'\_ and '\_ecod'\_ are both anagrams of \_'code'\_ so they are removed from the list. The words \_'frame'\_ and '\_framer'\_ are not anagrams due to the extra \_'r'\_ in \_'framer',\_ so they remain. The final list of strings in alphabetical order is \_['code', 'frame', 'framer'].\_

**Input**

A list of strings \_s\_ that meet the criteria above.

**Output**

A list of strings in alphabetical order, ascending.

**Expected Output**

Test 1

Input: 4 code aaagmnrs anagrams doce

Output: aaagmnrs code

Test 2

Input: 4 poke pkoe okpe ekop

Output: poke