Data Structures and Algorithms

Problem Set #4

Topics : Maps, Vectors, STL

Problems

1. **All About Duplicates**

   (a) Given an array, give a linear time algorithm to check whether the array contains duplicate elements or not.

   (b) Suppose that the entries of the array are numbers between 0 to $n-1$ where n is the length of the array. Give a linear time algorithm to check if it contains duplicates. You should use $O(1)$ extra space.

2. **Find Missing** : Given an array containing $n-1$ numbers ranging from 1 to n, each element appears exactly once. Hence, one element is missing. Give an algorithm to find the missing number. Start with $O(n^2)$, optimize it to $O(n \log n)$, and further to $O(n)$. Finally, come up with a linear time algorithm which uses constant space. Note that you should take care of overflows.

3. **First Repeating Element** : Given an array, devise a linear time algorithm to find the first repeating number in the that array.

4. **First Non Repeating Character** : Given a *string*, find the first non repeating character in that string. The expected time complexity is linear, while using O(1) space. An approach which uses 2 pass of the string is pretty obvious. Can you do it in a single pass while maintaining the restrictions of time and space complexity?
   Hint : Why does it matter that the input is a string?

5. **2 Sum** : Given an array of n elements, find 2 numbers whose sum is equal to k if they exist. Optimise your solution from $O(n^2) \longrightarrow O(n \log n) \longrightarrow O(n)$

6. **Permutations** : Given 2 array, write an algorithm to determine whether they are permutations of each other or not. Would sorting work? How would you apply bruteforce? How do you optimize it?

7. **Majority Element** : A *Majority element* is defined to be an element which appears more than $n/2$ times in an array. Give an optimal algorithm to find the majority element if it exists.

8. **The Meaning to Life** : Given a book, represented by strings, give an algorithm to find the most frequent word.

9. **Distinct Elements in Every Window of Size k :** You should really think on this question. If you can do this, it means that you've understood maps well. A window of size k is defined to be a contiguous sub array of length k. Given an array, find the number of distinct elements in every window of size k.