Leetcode Questions & Answers

ADITYA BYJU

"Something is useful if it behaves exactly as expected"

IIT Bombay

March 2023

Contents

1 Blind 75

1.1 Arrays and Hashing

1.1.1 Contains Duplicate (217) - Easy

Question: Given an integer array nums, return true if any value appears at least twice in the array, and return false if every element is distinct.
Approaches:

- can use a hashset to store the elements and check for duplicate elements
- can compare the number of elements in the list and its unordered_set version

C++ Code:

```
class Solution {
public:
    bool containsDuplicate(vector<int>& nums) {
        unordered_set<int> a;
        for(int i=0; i<nums.size(); ++i){
            if(a.find(nums[i])!=a.end()) return true;
            else a.insert(nums[i]);
        }
        return false;
    }
};
```

Python code:

```
class Solution:
    def containsDuplicate(self, nums: List[int]) -> bool:
        s=set()
        for x in nums:
            if x in s:
                return True
            else:
                s.add(x)
        return False
```

Time complexity: $O(n)$
Space complexity: $O(n)$
Remarks:

- In python set() is unordered set
- Function to add an element into a set() in python - add()

1.1.2 Valid Anagram (242) - Easy

Question: Given two strings s and t, return true if t is an anagram of s, and false otherwise.
An Anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.
Approaches:

- $O(nlogn)$ - compare the 2 sorted strings

- $O(n)$ - make an integer array of 26 zeroes, increment by one for the letters in the first word and decrement for the second one, check if all the elements in the array are zero

C++ Code:

```
class Solution {
public:
    bool isAnagram(string s, string t) {
        if(s.size()!=t.size()) return false;
        int a[26]={};
        for(int i=0; i<26; ++i) cout<<a[i]<<"\n";
        for(int i=0; i<s.size(); ++i){
            ++a[s[i]-'a'];
            --a[t[i]-'a'];
        }
        for(int i=0; i<26; ++i) if(a[i]!=0) return false;
        return true;
    }
};
```

Python code:

```
class Solution:
    def isAnagram(self, s: str, t: str) -> bool:
        a=[0]*26
        if len(s)!=len(t):
            return False
        for x in range(0, len(s)):
            a[ord(s[x])-ord('a')]+=1
            a[ord(t[x])-ord('a')]-=1
        for x in range(0,26):
            if a[x]!=0:
                return False
        return True
```

Time complexity: $O(n)$
Space complexity: $O(1)$
Remarks:

- In C++, an integer array can be intialized to NULL or 0 in the following ways:
  - int a[26]={};
  - int a[26]={0};
- In Python a list of zeroes can be created by: $a = [0]*26$
- In Python, the ord() function returns the number representing the unicode code of a specified character

1.1.3 Two Sum (1) - Easy

Question: Given an array of integers nums and an integer target, return indices of the two numbers such that they add up to target.

You may assume that each input would have exactly one solution, and you may not use the same element twice.

You can return the answer in any order.

Approaches:

- $O(n^2)$ - double loop iteration
- $O(n)$ - keep a map of elements and index, check if the (target-current) element is found in the map

C++ Code:

```
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        unordered_map<int, int> m;
        for(int i=0; i<nums.size(); ++i){
            if(m.find(target-nums[i])!=m.end()) return {m[target-nums[i]],i};
            else m[nums[i]] = i;
        }
        return {0,0};
    }
};
```

Python code:

```
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        d = {}
        for x in range(0, len(nums)):
            if target-nums[x] in d:
                return {d[target-nums[x]], x}
            else:
                d[nums[x]] = x
        return
```

Time complexity: $O(n)$
Space complexity: $O(n)$
Remarks:

- A map can be created in python by using {}
- In python, we don't have to return anything at the end because we know there exists a solution all the time

1.1.4 Group Anagrams (49) - Medium

Question: Given an array of strings strs, group the anagrams together. You can return the answer in any order.

An Anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

Approaches:

- Use the count of letters as key of a hashmap to group together anagrams

- Assign prime numbers to each of the 26 letters. Each anagram can be now equated to a unique number, which can act as the key of the hashmap.
- Can also sort the string to keep it as the key in the hashmap

C++ Code:

```
class Solution {
public:
    vector<vector<string>> groupAnagrams(vector<string>& strs) {
        unordered_map<string, vector<string>> a;
        vector<vector<string>> res;
        for(auto s: strs){
            string key = s;
            sort(key.begin(), key.end());
            a[key].push_back(s);
        }
        for(auto it = a.begin(); it!=a.end(); ++it){
            res.push_back(it->second);
        }
        return res;
    }
};
```

Python code:

```
class Solution:
    def groupAnagrams(self, strs: List[str]) -> List[List[str]]:
        grps = {}
        for string in strs:
            count = [0]*26
            for letter in string:
                count[ord(letter)-ord('a')]+=1
            key = '.'.join([str(n) for n in count])
            if key in grps:
                grps[key].append(string)
            else:
                grps[key] = [string]
        return grps.values()
```

Time complexity: $O(n.m)$, where n is the number of strings, and m is the length of a string
$O(n.m.log(m))$ if sorted string method is used
Space complexity: $O(n.m)$
Remarks:

- In Python, we can use .values() method on hashmap to return the values of the hashmap in a list.
- In Python .join() method can be used to join contents of a string list into a string using specfied delimiter.