COL106 Lab Week 9 Questions

Easy

Problem 1: Implement Trie

As you will soon realize, Trie is a very useful Data Structure. So, before we solve any problems, let's implement a trie ourselves. Your task is to implement the Trie data structure, as discussed in class.

You can submit and check your solution here.

Problem 2: Find the Town Judge

In a town, there are n people labelled from 1 to n. There is a rumour that one of these people is secretly the town judge.

If the town judge exists, then:

1. The town judge trusts nobody.
2. Everybody (except for the town judge) trusts the town judge.
3. There is exactly one person that satisfies properties 1 and 2.

You are given an array trust where trust[i] = [ai, bi] representing that the person labelled ai trusts the person labelled bi. If a trust relationship does not exist in the trust array, then such a trust relationship does not exist.

Return the label of the town judge if the town judge exists and can be identified, or return -1 otherwise.

You can submit and check your implementation here.

Medium

Problem 3: Deep Copy of a Graph

Given a graph, create a *deep copy* of it.

A deep copy means a completely independent copy of a data structure such that changes made to the copy do not affect the original, and vice versa. In contrast to a shallow copy which creates a new object that refers to the same underlying data as the original, a deep copy ensures that all levels of nested data structures within the object are also duplicated recursively.

You can submit and check your solution here.

Problem 4: Keys and Rooms

There are n rooms labelled from 0 to n - 1 and all the rooms are locked except for room 0. Your goal is to visit all the rooms. However, you cannot enter a locked room without having its key.

When you visit a room, you may find a set of distinct keys in it. Each key has a number on it, denoting which room it unlocks, and you can take all of them with you to unlock the other rooms.

Given an array “rooms” where rooms[i] is the set of keys that you can obtain if you visited room i, return true if you can visit all the rooms, or false otherwise.

Submit and check your code here.

(P.S. Graph traversals are important for this problem. You can read more about them here)

Hard

Problem 5: Maximum XOR-pair in an array

Given an (non-negative) integer array nums[], you need to find the pair of elements with maximum XOR value. (Recall that XOR is true iff exactly one of the two bits is true).

The naive approach is to consider all pairs of elements, compute XOR and take maximum. Convince yourself that the TC for this is O(N^2) (More accurately, there should be a factor of logM - where M is the maximum value of element in the array - but since M is bounded by say, 10^9, we can ignore logM).

Can you do better than this? Link for problem

Problem 6: Count XOR Pairs in Given Range

You are again given a non-negative integer array nums[]. You are also given a range [low, high]. Count the number of pairs, such that their XOR lies in this given range.

For a formal description, please see this.

Follow Up: Say you only given the array A of n integers, and an integer P. Define NP_x as the number of pairs of elements of array A, such that their bitwise xor is greater than or equal to X. Find maximum X such that NP_x >= P.