LEETCODE QUES

Ques 128

class Solution {

public:

    int longestConsecutive(vector<int>& nums) {

        unordered\_map<int, int> seqMap;

        int longest = 0;

        for(int num : nums){

            if(seqMap.find(num) == seqMap.end()){ // Ignore Duplicate

                     int left = seqMap.count(num - 1) ? seqMap[num-1]:0;

                     int right = seqMap.count(num + 1) ? seqMap[num+1]:0;

                     int length = left + right + 1;

                     seqMap[num] = length;

                     seqMap[num - left] = length; //Update Left Boundary

                     seqMap[num + right] = length; //Update Right Boundary

                     longest = max(longest, length);

            }

        }

        return longest;

    }

};

Ques 728

class MyHashSet {

private:

    int size;

    vector<list<int>> bucket;

    int hash(int key) {

        return key % size;

    }

public:

    MyHashSet() {

        size = 10007;

        bucket.resize(size);

    }

    void add(int key) {

        int index = hash(key);

        if (!contains(key)) {

            bucket[index].push\_back(key);

        }

    }

    void remove(int key) {

        int index = hash(key);

        bucket[index].remove(key);

    }

    bool contains(int key) {

        int index = hash(key);

        for (int num : bucket[index]) {

            if (num == key) {

                return true;

            }

        }

        return false;

    }

};

/\*\*

 \* Your MyHashSet object will be instantiated and called as such:

 \* MyHashSet\* obj = new MyHashSet();

 \* obj->add(key);

 \* obj->remove(key);

 \* bool param\_3 = obj->contains(key);

 \*/