Experiment 1

Name: Praduman Kumar

UID: 20BCS9446

Section/Group: DM 714-A

Subject: Competitive Coding II

Semester: 6th

Date of Performance: 17/02/2023

Aim:

Arrays, Stacks, Queues Linked List

Problem 1: 3Sum

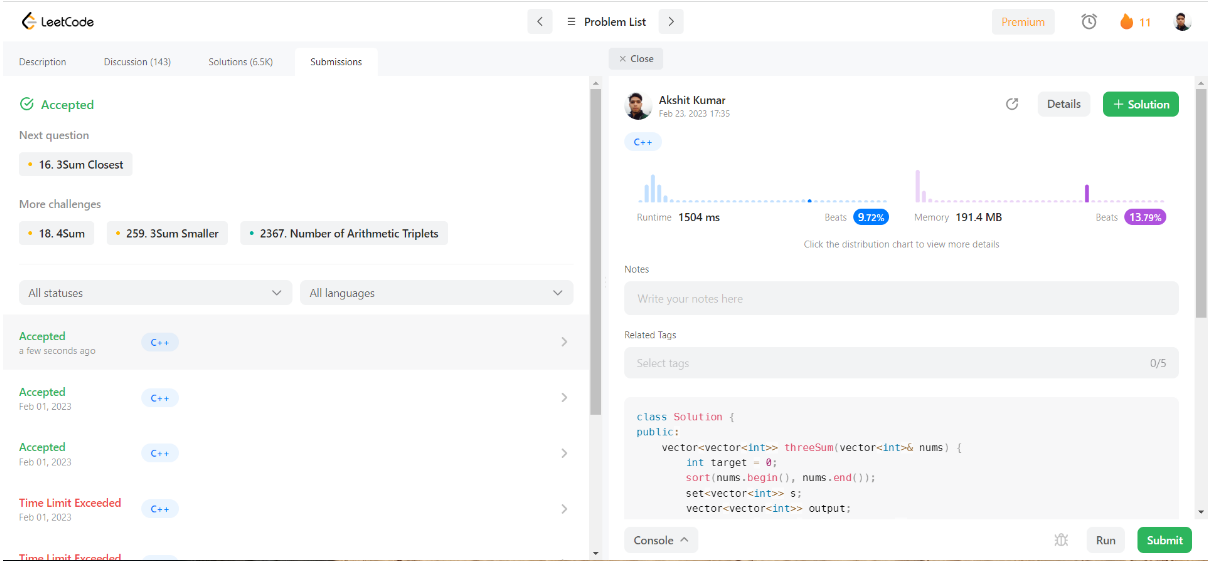
Given an integer array nums, return all the triplets [nums[i], nums[j], nums[k]] such that i != j, i != k, and j != k, and nums[i] + nums[j] + nums[k] == 0.

Notice that the solution set must not contain duplicate triplets.

Code:

```
class Solution {

public:

    vector<vector<int>> threeSum(vector<int>& nums) {

        int target = 0;

        sort(nums.begin(), nums.end());

        set<vector<int>> s;

        vector<vector<int>> output;

        for (int i = 0; i < nums.size(); i++){

            int j = i + 1;

            int k = nums.size() - 1;

            while (j < k) {
                int sum = nums[i] + nums[j] + nums[k];
                if (sum == target) {
                    s.insert({nums[i], nums[j], nums[k]});
                    j++;
                    k--;
                } else if (sum < target) {
                    j++;
                } else {
                    k--;
                }
            }
        }
        for(auto triplets : s)
            output.push_back(triplets);
        return output;
    }
};
```

Output:



Accepted

Next question

- 16. 3Sum Closest

More challenges

- 18. 4Sum
- 259. 3Sum Smaller
- 2367. Number of Arithmetic Triplets

Accepted C++
a few seconds ago

Accepted C++
Feb 01, 2023

Accepted C++
Feb 01, 2023

Time Limit Exceeded C++
Feb 01, 2023

Time Limit Exceeded

Akshit Kumar
Feb 23, 2023 17:35

C++

Runtime 1504 ms Beats 9.72%

Memory 191.4 MB Beats 13.79%

Click the distribution chart to view more details

```
class Solution {
public:
    vector<vector<int>> threeSum(vector<int>& nums) {
        int target = 0;
        sort(nums.begin(), nums.end());
        set<vector<int>> s;
        vector<vector<int>> output;
```

Problem 2: Jump Game II

You are given a 0-indexed array of integers nums of length n. You are initially positioned at nums[0].

Each element nums[i] represents the maximum length of a forward jump from the index i. In other words, if you are at nums[i], you can jump to any nums[i + j] where:

0 <= j <= nums[i] and

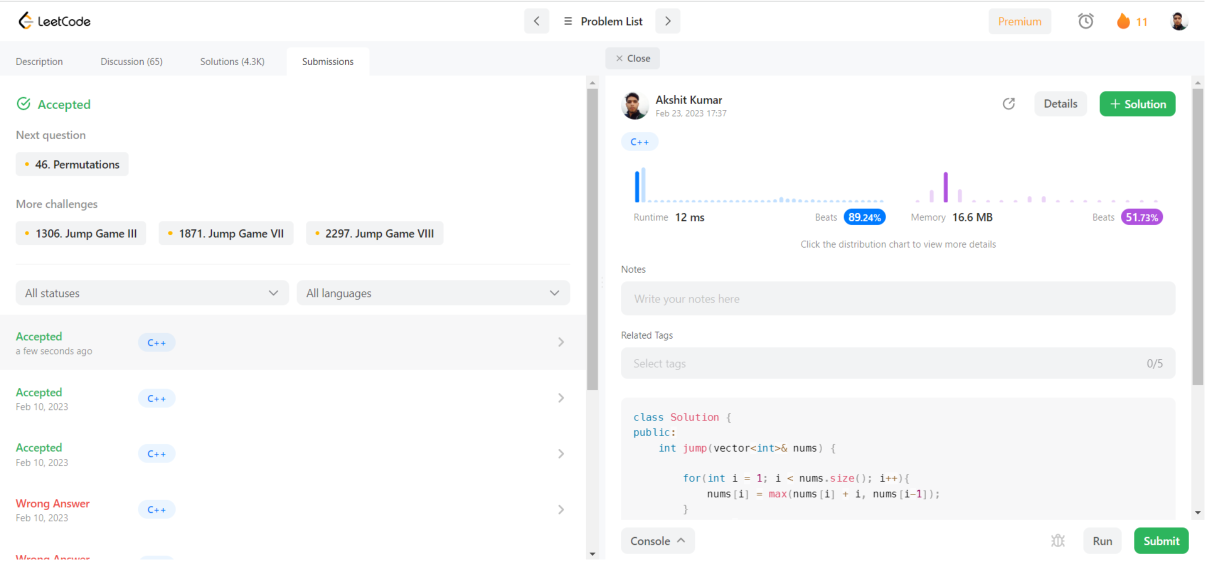
i + j < n

Return the minimum number of jumps to reach nums[n - 1]. The test cases are generated such that you can reach nums[n - 1].

Code:

```
class Solution {
public:
    int jump(vector<int>& nums) {

        for(int i = 1; i < nums.size(); i++){
            nums[i] = max(nums[i] + i, nums[i-1]);
        }
        int ans = 0;
        int i = 0;
        while(i<nums.size()-1){
            ans++;
            i = nums[i];
        }
        return ans;
    }
};
```

Output:



Accepted

Next question

46. Permutations

More challenges

1306. Jump Game III
1871. Jump Game VII
2297. Jump Game VIII

All statuses
All languages

Accepted a few seconds ago C++
Accepted Feb 10, 2023 C++
Accepted Feb 10, 2023 C++
Wrong Answer Feb 10, 2023 C++

Akshit Kumar
Feb 23, 2023 17:37

C++

Runtime 12 ms Beats 89.24%
Memory 16.6 MB Beats 51.73%

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags 0/5

```
class Solution {
public:
    int jump(vector<int>& nums) {

        for(int i = 1; i < nums.size(); i++){
            nums[i] = max(nums[i] + i, nums[i-1]);
        }
```