Experiment-2.2

Student Name: Aayush Gurung **UID:20BCS5323**
Branch: CSE **Section/Group:DM_607(A)**
Semester: 6 **Date of Performance:07-03-2023**
Subject Name: CC LAB **Subject Code: 20CSP-351**

Aim:

To demonstrate the concept of Graphs

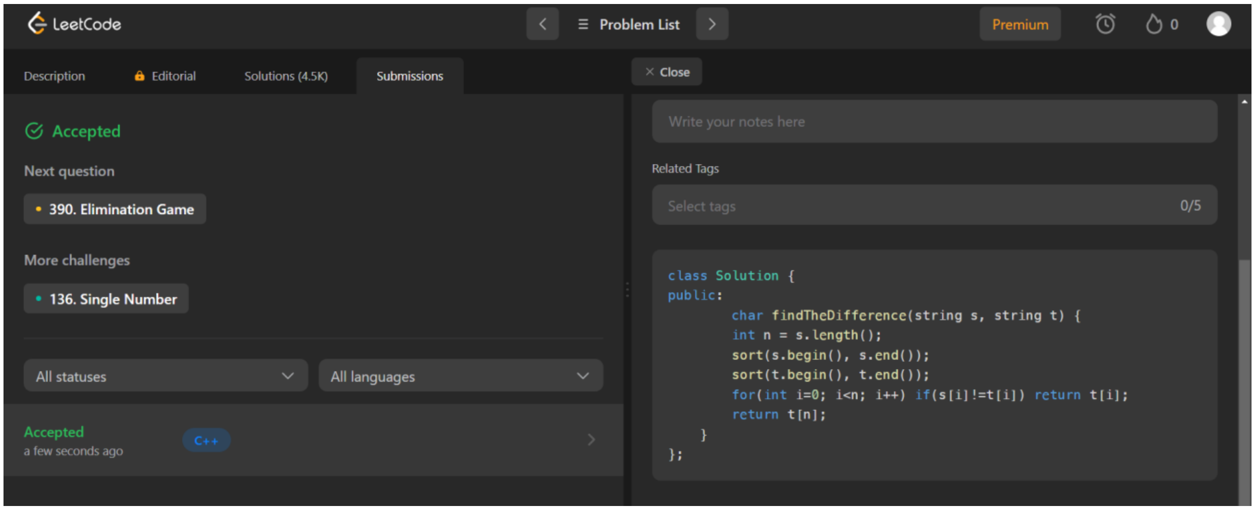
Objective

Problem 1:
Find the Difference

Code:

```
class Solution {

public:

        char findTheDifference(string s, string t) {

        int n = s.length();

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

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

        for(int i=0; i<n; i++) if(s[i]!=t[i]) return t[i];

        return t[n];

    }

};
```

Output:



Accepted

Accepted
a few seconds ago

```
class Solution {
public:
        char findTheDifference(string s, string t) {
        int n = s.length();
        sort(s.begin(), s.end());
        sort(t.begin(), t.end());
        for(int i=0; i<n; i++) if(s[i]!=t[i]) return t[i];
        return t[n];
    }
};
```

Problem 2:

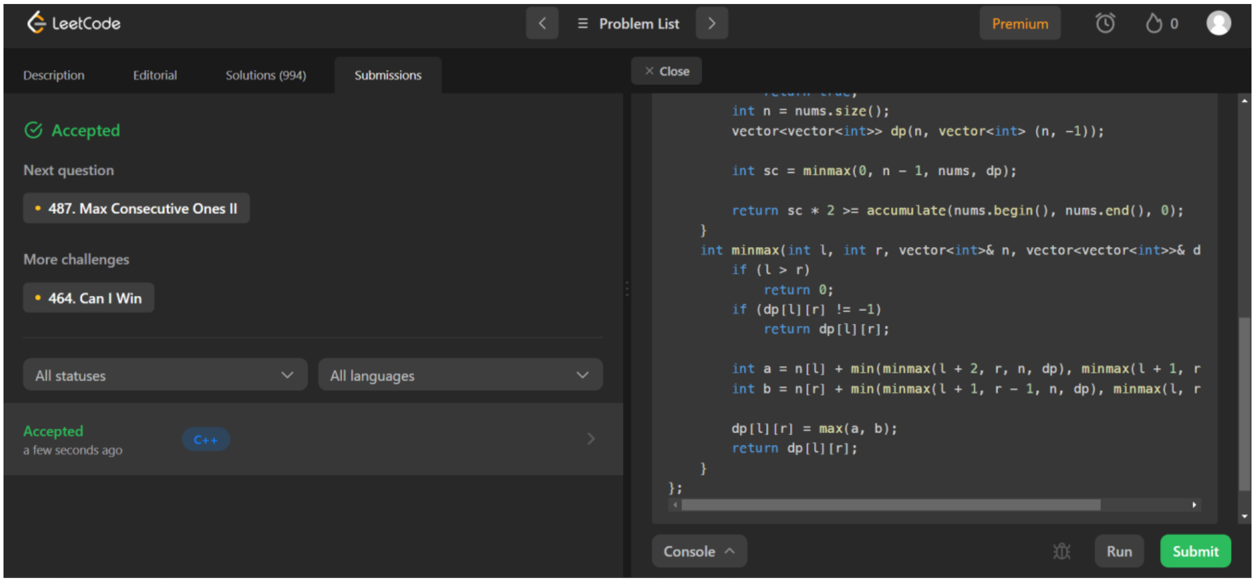
Predict the winner

Code:

```
class Solution {
public:
    bool PredictTheWinner(vector<int>& nums) {
        if (nums.size() <= 2)
            return true;
        int n = nums.size();
        vector<vector<int>> dp(n, vector<int> (n, -1));
        int sc = minmax(0, n - 1, nums, dp);
        return sc * 2 >= accumulate(nums.begin(), nums.end(), 0);
    }
    int minmax(int l, int r, vector<int>& n, vector<vector<int>>& dp){
        if (l > r)
            return 0;
        if (dp[l][r] != -1)
```

```
            return dp[l][r];
        int a = n[l] + min(minmax(l + 2, r, n, dp), minmax(l + 1, r - 1, n, dp));
        int b = n[r] + min(minmax(l + 1, r - 1, n, dp), minmax(l, r - 2, n, dp));
        dp[l][r] = max(a, b);
        return dp[l][r];
    }
};
```

Output:



```
            return true;
        int n = nums.size();
        vector<vector<int>> dp(n, vector<int> (n, -1));

        int sc = minmax(0, n - 1, nums, dp);

        return sc * 2 >= accumulate(nums.begin(), nums.end(), 0);
    }
    int minmax(int l, int r, vector<int>& n, vector<vector<int>>& d
        if (l > r)
            return 0;
        if (dp[l][r] != -1)
            return dp[l][r];

        int a = n[l] + min(minmax(l + 2, r, n, dp), minmax(l + 1, r
        int b = n[r] + min(minmax(l + 1, r - 1, n, dp), minmax(l, r

        dp[l][r] = max(a, b);
        return dp[l][r];
    }
};
```