DSA SERIES

- Learn Coding

Topic to be Covered today

Greedy Algorithms

948. Bag of Tokens

```
class Solution {
public:
    int bagOfTokensScore(vector<int>& tokens, int power) {
        int n = tokens.size();


        int maxScore = 0;
        sort(begin(tokens),end(tokens));


        int i = 0 , j = n-1;


        int score = 0;


        while(i<=j){
            if(power >= tokens[i]){
                power -=tokens[i];
                score++;
                i++;
```

```
                maxScore = max(maxScore ,score);
            }
            else if(score >= 1){
                power += tokens[j];
                score--;
                j--;
            }else{
                return maxScore;
            }
        }


        return maxScore;
    }
};
```

2405. Optimal Partition of String

```
class Solution {
public:
    int partitionString(string s) {
        int n = s.length();

        vector<int> lastSeen(26,-1);
        int count = 0;
        int curr = 0;

        for(int i =0;i<n;i++){
            char ch = s[i];

            if(lastSeen[ch-'a'] >= curr){
                    count++;
                    curr = i;
            }
            lastSeen[ch-'a']=i;
        }

        return count + 1;
    }
};
```

1578. Minimum Time to Make Rope Colorful

```
class Solution {
public:
    int minCost(string colors, vector<int>& neededTime) {
        int m = colors.length();


        int time = 0;


        int prev = 0;
        int curr = 1;


        while (curr < m) {
            if (colors[prev] == colors[curr]) {
                int val1 = neededTime[prev];
                int val2 = neededTime[curr];


                if (val1 > val2) {
                    time += val2;
```

```
            curr++;
                } else {
                    prev = curr;
                    time += val1;
                    curr++;
                }
            } else {
                prev = curr;
                curr++;
            }
        }


        return time;
    }
};
```

134. Gas Station

```
class Solution {
public:
    int canCompleteCircuit(vector<int>& gas, vector<int>& cost) {
        int n = gas.size();


        int totalEarn = accumulate(begin(gas),end(gas),0);
        int totalExpend = accumulate(begin(cost),end(cost),0);


        if(totalEarn < totalExpend){
            return -1;
        }



        int result = 0;
        int total = 0;
```

```
        for(int i =0;i<n;i++){
            total += gas[i] - cost[i];


            if(total < 0){
                result = i+1;
                total =0;
            }
        }


        return result;
    }
};
```

2038. Remove Colored Pieces if Both Neighbors are the Same Color

```
class Solution {
public:
    bool winnerOfGame(string colors) {
        int n = colors.length();


        int alice = 0;
        int bob = 0;


        for (int i = 1; i < n - 1; i++) {
            if ((colors[i - 1] == colors[i]) && (colors[i] == colors[i + 1])) {
                if (colors[i] == 'A') {
                    alice++;
                } else {
                    bob++;
                }
            }
        }
        return alice > bob;
    }
};
```

Learn coding

THANK YOU