July 14, 2018

346

1. 毕竟Easy

```
class MovingAverage {
public:
    /** Initialize your data structure here. */
    int cnt, cap, sum;
    queue<int> Q;
    MovingAverage(int size): cap(size), cnt(0), sum(0){}

    double next(int val) {
        Q.push(val);
        sum += val;
        if(cnt<cap) ++cnt;
        else{
            sum -= Q.front();
            Q.pop();
        }
        return double(sum)/cnt;
    }
};
```

832

1. 居然坐错一次，边界没搞对 “`(i < (vec.size()+1)/2)`"

```
class Solution {
public:
    vector<vector<int>> flipAndInvertImage(vector<vector<int>>
& A) {
        for(auto &vec: A) {
            for(int i=0; i<(vec.size()+1)/2; ++i){
                int tmp = vec[i];
                vec[i] = 1 - vec[vec.size()-i-1];
                vec[vec.size()-i-1] = 1 - tmp;
            }
        }
        return A;
    }
};
```

408

1. 注意二逼corner case：`n == 0`, 或者最后 数字加完长度超过原长。

```
class Solution {
public:
    bool validWordAbbreviation(string word, string abbr) {
        int i=0, j=0;
        while(i<word.size() && j<abbr.size()){
            if(isdigit(abbr[j])){
                int cnt = stoi(abbr.substr(j));
                if(!cnt) return false;
                i += cnt;
                j += (int)to_string(cnt).size();
            }
            else if(word[i++] != abbr[j++]) return false;
        }
        return (i==word.size() && j==abbr.size());
    }
};
```

85

1. O(n * m)

```
class Solution {
    int max1DRec(vector<int> &h){
        // h[0] = -2, h[h.size()-1] = -1;
        stack<int> S;
        int ans = 0;
        S.push(0);
        for(int i=1; i<h.size(); ++i){
            while(h[S.top()] >= h[i]){
                int H = h[S.top()];
                S.pop();
                ans = max(ans, H * (i-1-S.top()));
            }
            S.push(i);
        }
        return ans;
    }
public:
    int maximalRectangle(vector<vector<char>>& matrix) {
        if(matrix.empty() || matrix[0].empty()) return 0;
        int n = matrix.size(), m = matrix[0].size(), ans = 0;
        vector<int> dp(m+2, 0);
        dp[0] = -2;
        dp[m+1] = -1;
        for(int i=0; i<n; ++i){
            for(int j=0; j<m; ++j){
                if(matrix[i][j] == '1') dp[j+1] += 1;
                else dp[j+1] = 0;
            }
            ans = max(ans, max1DRec(dp));
        }
        return ans;
    }
};
```

258

1. Recursion:

```
class Solution {
public:
    int addDigits(int num) {
        if(num < 10) return num;
        if(num %9 == 0) return 9;
        int n = 0;
        while(num){
            n += num%10;
            num /= 10;
        }
        return addDigits(n);
    }
};
```

2. 讨论区真是大神如云：

```
class Solution {
public:
    int addDigits(int num) {
        if(num<10) return num;
        return (num%9==0?9:(num%9));
    }
};
```

830

1. 毕竟Easy

```
class Solution {
public:
    vector<vector<int>> largeGroupPositions(string S) {
        vector<vector<int>> ans;
        S += "#";
        for(int i=1, cnt=1; i<S.size(); ++i){
            if(S[i]==S[i-1]) ++cnt;
            else{
                if(cnt > 2)ans.push_back(vector<int>{i-cnt, i-
1});
                cnt = 1;
            }
        }
        return ans;
    }
};
```

848 (直接搜搜不到)

1. 简单难度的Medium，但是注意别看错题

```
class Solution {
public:
    string shiftingLetters(string S, vector<int>& shifts) {
        for(int i=S.size()-1, pos=0;i>=0;--i){
            pos = (pos + shifts[i]) % 26;
            S[i] = char('a' + (int(S[i]-'a') + pos)%26);
        }
        return S;
    }
};
```

1. 容斥原理计数问题

```
class Solution {
public:
    int numSubarrayBoundedMax(vector<int>& A, int L, int R) {
        A.push_back(R+1);
        int i = 0, ans = 0;
        while(i < A.size()){
            int tmp = 0, cnt = 0, n = 0;
            while(i<A.size() && A[i]<=R){
                if(A[i] < L) ++cnt;
                else {
                    tmp += cnt * (cnt + 1) / 2;
                    cnt = 0;
                }
                ++i;
                ++n;
            }
            tmp += cnt * (cnt + 1) / 2;
            ans += n * (n+1)/2 - tmp;
            ++i;
        }
        return ans;
    }
};
```

511

empty

167

1. 一天Easy，真实轻松

```
class Solution {
```

```
public:
    vector<int> twoSum(vector<int>& numbers, int target) {
        int i = 0, j = numbers.size() - 1;
        while(i < j){
            if(numbers[i] + numbers[j] < target) ++i;
            else if(numbers[i] + numbers[j] > target) --j;
            else return vector<int>{i+1, j+1};
        }
        return vector<int>{-1, -1};
    }
};
```