August 8, 2018 题目：345,782,480,356,845,823,530,380,145,555,849,41,583

345

1. One pass (`Macro > constants > variables`):

```
class Solution {
    #define ISVOW(c) (((c)=='a') || ((c)=='e') || bool((c)=='i') || bool((c)=='o') || bool((c)=='u') || bool((c)=='A') || bool((c)=='E') || bool((c)=='I') || bool((c)=='O') || bool((c)=='U'))
public:
    string reverseVowels(string s) {
        for(int i=0, j=s.size()-1; i<j; ++i, --j){
            while(i<j && !ISVOW(s[i])) ++i;
            while(i<j && !ISVOW(s[j])) --j;
            swap(s[i], s[j]);
        }
        return s;
    }
};
```

782

1. 题目完全是用来吓唬人的，其实超简单：就是对第一行跟第一列分别操作，操作完看其他行列是否满足 `1` `0`相间即可。

```
class Solution {
    typedef vector<int> vi;
    bool isChess(vector<vector<int>>& B){
        int n = B.size();
        vi O(n, 0), E(n, 0);
        for(int i=0; i<n; ++i){
            if(i%2) O[i] = 1;
            else E[i] = 1;
        }
        if(B[0] != O) swap(E, O);
        for(int j=0; j<n; ++j){
            if(j%2 && B[j]!=E) return false;
            if(j%2==0 && B[j]!=O) return false;
        }
        return true;
    }
public:
    int movesToChessboard(vector<vector<int>>& B) {
        int n = B.size(), ans = 0;
        vi e1, e0, o1, o0;
        for(int j=0; j<n; ++j){
            if(j%2){
                if(B[0][j]) o1.push_back(j);
                else o0.push_back(j);
            }
            else{
                if(B[0][j]) e1.push_back(j);
                else e0.push_back(j);
            }
        }
        if(e1.size()!=o0.size() && e0.size()!=o1.size()) retur
n -1;
        if(e0.size()==o1.size() && (e1.size()!=o0.size() || e
1.size()>e0.size())){
            swap(e1, o1);
            swap(e0, o0);
        }
        for(int k=0; k<e1.size(); ++k) for(int i=0; i<n; ++i)
swap(B[i][e1[k]], B[i][o0[k]]);
        ans += e1.size();
        e1.clear();
        e0.clear();
        o1.clear();
        o0.clear();
        for(int j=0; j<n; ++j){
            if(j%2){
                if(B[j][0]) o1.push_back(j);
                else o0.push_back(j);
            }
            else{
                if(B[j][0]) e1.push_back(j);
                else e0.push_back(j);
            }
        }
        if(e1.size()!=o0.size() && e0.size()!=o1.size()) retur
n -1;
        if(e0.size()==o1.size() && (e1.size()!=o0.size() || e
1.size()>e0.size())){
            swap(e1, o1);
            swap(e0, o0);
        }
        for(int k=0; k<e1.size(); ++k) swap(B[e1[k]], B[o0
[k]]);
        ans += e1.size();
        if(isChess(B)) return ans;
        return -1;
```

```
    }
};
```

480

1. max Heap & min heap; same as 295

```
class Solution {
    public double[] medianSlidingWindow(int[] nums, int k) {
        double[] res = new double[nums.length - k + 1];

        PriorityQueue<Integer> min = new PriorityQueue<>();
        PriorityQueue<Integer> max = new PriorityQueue<>(1, Co
llections.reverseOrder());
        int j = 0;
        for (int i = 0; i < nums.length; i++) {
            max.offer(nums[i]);
            min.offer(max.poll());
            if (min.size() > max.size()) {
                max.offer(min.poll());
            }
            if (max.size() + min.size() == k) {
                if (max.size() == min.size()) {
                    res[j] = ((long)max.peek() + (long)min.pee
k()) / 2.0;
                } else {
                    res[j] = max.peek();
                }
                if (!max.remove(nums[j])) {
                    min.remove(nums[j]);
                }
                j++;
            }
        }
        return res;
    }
}
```

2. c++ 连个正对这的multiset:
    - 跟295不同的是，这个数据结构要能remove，所以改用multiset.

```
class Solution {
    struct Medusa{
        multiset<long> upper;
        multiset<long, greater<long>> lower;
        Medusa(){}
        void adjust(){
            if(upper.size() > lower.size()+1){
                lower.insert(*upper.begin());
                upper.erase(upper.begin());
            }
            else if(lower.size() > upper.size()){
                upper.insert(*lower.begin());
                lower.erase(lower.begin());
            }
        }
        void insert(long k){
            if(upper.empty()){
                upper.insert(k);
            }
            else if(k >= *upper.begin()) upper.insert(k);
            else lower.insert(k);
            this->adjust();
        }
        void remove(long k){
            if(upper.count(k)) upper.erase(upper.find(k));
            else if(lower.count(k)) lower.erase(lower.find(k));
            this->adjust();
        }
        double getMed(){
            if(upper.size() == lower.size()){
                return 0.5 * (*upper.begin() + *lower.begin());
            }
            else return double(*upper.begin());
        }
    };
public:
    vector<double> medianSlidingWindow(vector<int>& nums, int k) {
        Medusa med;
        vector<double> ans;
        for(int i=0; i<k-1; ++i) med.insert(nums[i]);
        for(int i=k-1;i<nums.size();++i){
            med.insert(nums[i]);
            ans.push_back(med.getMed());
            med.remove(nums[i-k+1]);
        }
        return ans;
    }
};
```

356

1. 按 `y` 的值列好就行了:

```
class Solution {
```

```
public:
    bool isReflected(vector<pair<int, int>>& points) {
        if(points.empty()) return true;
        unordered_map<long, set<long>> pos;
        unordered_map<long, vector<long>> npos;
        for(auto p: points) pos[(long)p.second].insert(long(p.first));
        for(auto p: pos) npos[p.first] = vector<long>(p.second.begin(), p.second.end());
        pos.clear();
        long cur = npos.begin()->second[0] + npos.begin()->second.back();
        for(auto p: npos){
            int n = p.second.size();
            for(int i=0, j=n-1; i<=j ; ++i, --j) if(p.second[i] + p.second[j] != cur) return false;
        }
        return true;
    }
};
```

2. Improved, `O(nlogn)` Time & `O(1)` space, learned from LeetCode discussion:

```
class Solution {
    typedef pair<int, int> ii;
public:
    bool isReflected(vector<pair<int, int>>& points) {
        int n = points.size();
        if(points.empty()) return true;
        long m = long(INT_MAX), M = long(INT_MIN);
        for(auto p: points){
            m = min(m, long(p.first));
            M = max(M, long(p.first));
        }
        long cur = m + M;
        sort(points.begin(), points.end(), [cur](ii a, ii b){
            if(a.first == b.first){
                if(long(a.first) * 2 < cur) return a.second <
b.second;
                else return a.second > b.second;
            }
            return a.first < b.first;
        });
        int i = 0, j = n-1;
        ii head = points[i], tail = points[j];
        while(i<=j){
            if(long(head.first) + long(tail.first) != cur) ret
urn false;
            if(2 * long(head.first) != cur && head.second != t
ail.second) return false;
            while(i<=j && points[i]==head) ++i;
            head = points[i];
            while(j>=i && points[j]==tail) --j;
            tail = points[j];
        }
        return true;
    }
};
```

- [ ] Some interesting Ideas: https://leetcode.com/problems/line-reflection/discuss/155751/5ms-beat-96.74.-one-pass-O(nlogn)-time-complexity-O(1)-space-complexity @Zebo L

845

1. Easy one pass:

```
class Solution {
```

```
public:
    int longestMountain(vector<int>& nums) {
        int n = nums.size(), i = 0, ans = 0;
        while(i<n-1){
            while(i<n-1 && nums[i]>=nums[i+1]) ++i;
            int j = i;
            while(j<n-1 && nums[j]<nums[j+1]) ++j;
            if(j==n-1) break;
            if(nums[j]==nums[j+1]){
                i=j+1;
                continue;
            }
            while(j<n-1 && nums[j]>nums[j+1]) ++j;
            ans = max(j-i+1, ans);
            i = j;
        }
        return ans;
    }
};
```

823

1. 开始看错题了，以为root 是所有 subtree node 的乘积。愿题要相对简单些，直接dp即可，下面的方法`O(n^2)`,好像很慢：

```
class Solution {
    const long M = 1000000007;
    inline long ad(long x, long y){
        return (x + y) % M;
    }
    inline long mu(long x, long y){
        return (x * y) % M;
    }
    vector<long> dp, A;
    unordered_map<long, int> pos;
public:
    int numFactoredBinaryTrees(vector<int>& Aa) {
        sort(Aa.begin(), Aa.end());
        long n = (long)Aa.size(), ans = 0;
        dp = vector<long>(n, 1L);
        A = vector<long>(n, 0L);
        for(int i=0; i<n; ++i) A[i] = long(Aa[i]);
        for(int i=0; i<n ;++i) pos[A[i]] = i;
        for(int j=1; j<n; ++j){
            for(int i=0; i<j; ++i) if(A[j]%A[i]==0&&pos.count
(A[j]/A[i])) {
                dp[j] = ad(dp[j], mu(dp[i], dp[pos[A[j]/A
[i]]]));
            }
        }
        for(long k: dp) ans = ad(ans, k);
        return ans % M;
    }
};
```

2. 下面这个选自 LeetCode Discussion，简洁而且快：

```
class Solution {
public:
    int numFactoredBinaryTrees(vector<int>& A) {
        long res = 0, mod = pow(10, 9) + 7;
        sort(A.begin(), A.end());
        unordered_map<int, long> dp;
        for (int i = 0; i < A.size(); ++i) {
            dp[A[i]] = 1;
            for (int j = 0; j < i; ++j)
```

```
                if (A[i] % A[j] == 0 && dp.count(A[i] / A[j])
== 1)
                    dp[A[i]] = (dp[A[i]] + dp[A[j]] * dp[A[i]
/ A[j]]) % mod;
        }
        for (auto &it : dp) res = (res + it.second) % mod;
        return res;
    }
};
```

530

1. Inorder Traversal:

```
class Solution {
public:
    int getMinimumDifference(TreeNode* root) {
        long dif = -1, cur = 0;
        stack<TreeNode *> S;
        while(root || !S.empty()){
            while(root){
                S.push(root);
                root = root->left;
            }
            if(!S.empty()){
                if(dif == -1) dif = long(2) * INT_MAX;
                else dif = min(dif, (long)S.top()->val - cur);
                cur = (long)S.top()->val;
                root = S.top()->right;
                S.pop();
            }
        }
        return dif;
    }
};
```

380

1. Map 都做过了，就是一个array 加 一个 hash map：

```
class RandomizedSet {
public:
    vector<int> A;
    unordered_map<int, int> B;
    RandomizedSet() {}
    bool insert(int val) {
        if(B.count(val)) return false;
        B[val] = A.size();
        A.push_back(val);
        return true;
    }
    bool remove(int val) {
        if(!B.count(val)) return false;
        if(A.back() != val) A[B[A.back()]=B[val]] = A.back();
        B.erase(val);
        A.pop_back();
        return true;
    }
    int getRandom() { return A[rand()%(int)A.size()]; }
};
```

145

1. Reverse preorder

```
class Solution {
public:
    vector<int> postorderTraversal(TreeNode* root) {
        stack<TreeNode *> S;
        vector<int> ans;
        while(!S.empty() || root){
            while(root){
                ans.push_back(root->val);
                if(root->left) S.push(root->left);
                root = root->right;
            }
            if(!S.empty()){
                root = S.top();
                S.pop();
            }
        }
        reverse(ans.begin(), ans.end());
        return ans;
    }
};
```

555

1. Easy brute force:

```
class Solution {
public:
    string splitLoopedString(vector<string>& strs) {
        string ans, res;
        for(string &s: strs){
            string tmp = s;
            reverse(tmp.begin(), tmp.end());
            res += max(tmp, s);
        }
        for(int i=0, cur=0; i<strs.size(); cur+=strs[i++].size
()){
```

```
            string tmp = res.substr(cur+strs[i].size()) + res.
substr(0, cur), s = strs[i];
            for(int j=0; j<s.size(); ++j) ans = max(ans, s.sub
str(j) + tmp + s.substr(0, j));
            reverse(s.begin(), s.end());
            for(int j=0; j<s.size(); ++j) ans = max(ans, s.sub
str(j) + tmp + s.substr(0, j));
        }
        return ans;
    }
};
```

849

1. 注意两边即可：

```
class Solution {
public:
    int maxDistToClosest(vector<int>& seats) {
        int ans = 0, cnt = 0, i = 0;
        while(i<seats.size() && !seats[i]) ++i;
        ans = i;
        for(auto n: seats){
            if(n) {
                ans = max(ans, (cnt-1)/2+1);
                cnt = 0;
            }
            else ++cnt;
        }
        return max(ans, cnt);
    }
};
```

41

1. swap to the position where nums[i] == i + 1

```
class Solution {
    public int firstMissingPositive(int[] nums) {
        int i = 0;
        while (i < nums.length) {
            if (nums[i] == i + 1 || nums[i] <= 0 || nums[i] > nums.length) i++;
            else if (nums[nums[i] - 1] != nums[i]) swap(nums, i, nums[i] - 1);
            else i++;
        }
        i = 0;
        while (i < nums.length && nums[i] == i + 1) i++;
        return i + 1;
    }

    public void swap(int[] nums, int i, int j) {
        int t = nums[i];
        nums[i] = nums[j];
        nums[j] = t;
    }
}
```

2. 注意边条件：

```
class Solution {
public:
    int firstMissingPositive(vector<int>& nums) {
        for(int i=0; i<nums.size(); ){
            if(nums[i]<=0 || nums[i]>nums.size() || nums[i]==i+1 || nums[i]==nums[nums[i]-1]) ++i;
            else swap(nums[i], nums[nums[i]-1]);
        }
        for(int i=0; i<nums.size(); ++i) if(nums[i] != i+1) re
turn i+1;
        return nums.size()+1;
    }
};
```

583

1. DP

```
class Solution {
    public int minDistance(String word1, String word2) {
        int[][] dp = new int[word1.length() + 1][word2.length
() + 1];
        dp[0][0] = 0;
        for (int i = 1; i < dp.length; i++) {
            for (int j = 1; j < dp[0].length; j++) {
                if (word1.charAt(i - 1) == word2.charAt(j -
1)) {
                    dp[i][j] = 1 + dp[i - 1][j - 1];
                } else {
                    dp[i][j] = Math.max(dp[i - 1][j], dp[i][j
- 1]);
                }
            }
        }
        int max = dp[word1.length()][word2.length()];
        return word1.length() - max + word2.length() - max;
    }
}
```

2. 同上：

```
class Solution {
```

```
public:
    int minDistance(string word1, string word2) {
        int n = word1.size(), m = word2.size();
        vector<int> dp(m+1, 0);
        for(int j=1; j<=m; ++j) dp[j] = j;
        for(int i=1; i<=n; ++i) {
            vector<int> tmp(dp);
            dp[0] = i;
            for(int j=1; j<=m; ++j){
                if(word1[i-1] == word2[j-1]) dp[j] = tmp[j-1];
                else dp[j] = 1 + min(tmp[j], dp[j-1]);
            }
        }
        return dp[m];
    }
};
```