August 9, 2018 题目：309,162,54,271,259,262,523,532,577,755,547,65,249

309

1. 注意最后计算结果的时候有两个 position 为 0 的状态

```
class Solution {
public:
    int maxProfit(vector<int>& prices) {
        int p1=INT_MIN, p2=INT_MIN, p0 = 0;
        for(int k: prices){
            int q1 = max(p1, p0-k), q0 = max(p2, p0), q2 = p1
+ k;
            p1 = q1;
            p2 = q2;
            p0 = q0;
        }
        return max(p0, p2);
    }
};
```

162

1. 二分，每次可以估测某个区间一定存在peak：

```
class Solution {
public:
    int findPeakElement(vector<int>& nums) {
        int n = nums.size();
        if(n==1) return 0;
        if(nums[0] > nums[1]) return 0;
        if(nums[n-1] > nums[n-2]) return n-1;
        int l = 1, r = n - 2;
        while(l < r){
            int c = (l+r)/2;
            if(nums[c] > nums[c+1]) r = c;
            else l = c+1;
        }
        return r;
    }
};
```

54

1. Easy Simulation:

```
class Solution {
public:
    vector<int> spiralOrder(vector<vector<int>>& matrix) {
        vector<int> ans;
        if(matrix.empty() || matrix[0].empty()) return ans;
        int n = matrix.size(), m = matrix[0].size();
        for(int i=0, j=0, k=0; k<n*m; ++k){
            ans.push_back(matrix[i][j]);
            if(i<=(n-1)/2 && i<=j+1 && i<(m-j-1)) ++j;
            else if(j>=m/2 && (m-j-1) <=i && (m-j-1) < (n-i-
1)) ++i;
            else if(i>(n-1)/2 && (n-i-1)<=(m-j-1) && (n-i-1) <
j) --j;
            else --i;
        }
        return ans;
    }
};
```

271

1. 跟string 有关的用python 就简单很多：

```
class Codec:
    def encode(self, strs):
        return ','.join([str(len(strs))] + [str(len(s)) for s in strs]) + ',' + ''.join(strs)
    def decode(self, s):
        tokens = s.split(',');
        n = int(tokens[0])
        if not n:
            return []
        temp = ','.join(tokens[n+1:])
        lens = [int(tokens[i+1]) for i in range(n)]
        assert sum(lens) == len(temp)
        ans = []
        cur = 0
        for l in lens:
            ans.append(temp[int(cur): int(cur+l)])
            cur += l
        return ans
```

259

1. 带两个指针的Brute Force：

```
class Solution {
public:
    int threeSumSmaller(vector<int>& nums, int target) {
        int n = nums.size(), ans = 0;
        if(n < 3) return 0;
        sort(nums.begin(), nums.end());
        for(int i=0; i<n-2 && nums[i]+nums[i+1]+nums[i+2]<target; ++i){
            int j = i+1, k = n-1;
            while(j<k){
                while(k>j && nums[k]>=target-nums[i]-nums[j]) --k;
                ans += k-j++;
            }
        }
        return ans;
    }
};
```

262

hard SQL

523

1. `mod k`就行了, `k=0` 的q情况比较恶心

```
class Solution {
public:
    bool checkSubarraySum(vector<int>& nums, int k) {
        if(!k) {
            for(int i=1; i<nums.size(); ++i) if(!nums[i]&&!nums[i-1]) return true;
            return false;
        }
        vector<int> P(nums.size()+1, 0);
        for(int i=1; i<=nums.size(); ++i) P[i] = (P[i-1] + nums[i-1])%k;
        unordered_set<int> S{P[0]};
        for(int i=2; i<=nums.size(); ++i){
            if(S.count(P[i])) return true;
            S.insert(P[i-1]);
        }
        return false;
    }
};
```

532

1. `k<0` 这种二B case 都能出出来：

```
class Solution {
public:
    int findPairs(vector<int>& nums, int k) {
        if(k<0) return 0;
        unordered_map<int, int> cnt;
        for(int k: nums) ++cnt[k];
        int ans = 0;
        for(auto p: cnt){
            if(k) ans += cnt.count(p.first + k);
            else ans += p.second > 1;
        }
        return ans;
    }
};
```

577

SQL

755

1. 分别maintain 左右区间 `O(V * log(N))`

```
class Solution {
public:
    vector<int> pourWater(vector<int>& H, int V, int K) {
        int l = K-1, r = K+1, n = H.size();
        map<int, set<int, greater<int>>> left;
        map<int, set<int>> right;
        while(l>=0 && H[l]<=H[l+1]){
            left[H[l]].insert(l);
            --l;
        }
        while(r<n && H[r]<=H[r-1]){
            right[H[r]].insert(r);
            ++r;
        }
        for(; V; --V){
            if(!left.empty() && H[K] > left.begin()->first){
                auto it = left.begin();
                int h = it->first, idx = *it->second.begin();
                left[h+1].insert(idx);
                it->second.erase(idx);
                if(it->second.empty()) left.erase(h);
                ++H[idx];
            }
            else if(!right.empty() && H[K] > right.begin()->first){
                auto it = right.begin();
                int h = it->first, idx = *it->second.begin();
                right[h+1].insert(idx);
                it->second.erase(idx);
                if(it->second.empty()) right.erase(h);
                ++H[idx];
            }
            else ++H[K];
            while(l>=0 && H[l]<=H[l+1]){
                left[H[l]].insert(l);
                --l;
            }
            while(r<n && H[r]<=H[r-1]){
                right[H[r]].insert(r);
                ++r;
            }
        }
        return H;
    }
};
```

2. An approach from https://leetcode.com/problems/pour-water/discuss/113003/C++JavaPython-O(V+N)-time-solution-using-2-pointers-and-2-stacks:
    - `O(V+N)`：茅塞顿开

```
class Solution {
public:
    vector<int> pourWater(vector<int>& H, int V, int K) {
        int l = K-1, r = K+1, n = H.size();
        stack<int> lfall, rfall;
        while(l>=0 && H[l]<=H[l+1]){
            if(H[l]<H[l+1]) lfall.push(l);
            --l;
        }
        while(r<n && H[r]<=H[r-1]){
            if(H[r]<H[r-1]) rfall.push(r);
            ++r;
        }
        while(V){
            if(!lfall.empty()){
                int idx = lfall.top();
                ++H[idx];
                if(H[idx]==H[idx+1]) lfall.pop();
                if(idx>l+1 && H[idx]>H[idx-1]) lfall.push(idx-
1);
            }
            else if(!rfall.empty()){
                int idx = rfall.top();
                ++H[idx];
                if(H[idx]==H[idx-1]) rfall.pop();
                if(idx<r-1 && H[idx]>H[idx+1]) rfall.push(idx+
1);
            }
            else{
                int idx = K;
                ++H[idx];
                if(idx>l+1 && H[idx]>H[idx-1]) lfall.push(idx-
1);
                if(idx<r-1 && H[idx]>H[idx+1]) rfall.push(idx+
1);
            }
            while(l>=0 && H[l]<=H[l+1]){
            if(H[l]<H[l+1]) lfall.push(l);
            --l;
            }
            while(r<n && H[r]<=H[r-1]){
                if(H[r]<H[r-1]) rfall.push(r);
                ++r;
            }
            --V;
        }
        return H;
```

```
    }
};
```

547

1. Union Find

```
class Solution {
    vector<int> P;
    int findRoot(int i){
        if(P[i]==i) return i;
        return P[i] = findRoot(P[i]);
    }
public:
    int findCircleNum(vector<vector<int>>& M) {
        int N = M.size(), ans = 0;
        P.resize(N);
        for(int i=0;i<N;++i) P[i] = i;
        for(int i=0; i<N; ++i) for(int j=i+1; j<N; ++j){
            if(M[i][j] && findRoot(i) != findRoot(j)){
                P[findRoot(j)] = findRoot(i);
            }
        }
        for(int i=0; i<N; ++i) ans += P[i]==i;
        return ans;
    }
};
```

2. UnionFind

```
class Solution {
    class UnionFind {
        int count;
        int[] parents;
        public UnionFind(int n) {
            parents = new int[n];
            for (int i = 0; i < n; i++) {
                parents[i] = i;
            }
            count = n;
        }
        public int find(int i) {
            while (parents[i] != i) {
                i = parents[i];
            }
            return i;
        }

        public void union(int i, int j) {
            int ii = find(i);
            int jj = find(j);
            if (ii != jj) {
                parents[ii] = jj;
                count--;
            }
        }
    }
    public int findCircleNum(int[][] M) {
        UnionFind uf = new UnionFind(M.length);
        for (int i = 0; i < M.length; i++) {
            for (int j = 0; j < M.length; j++) {
                if (M[i][j] == 1) {
                    uf.union(i, j);
                }
            }
        }
        return uf.count;
    }
}
```

65

1. 利用状态转换关系：

```
class Solution {
    int stat[9][6] = {
        {0,  1,  3,  2, -1, -1},
        {8,  1,  4, -1,  5, -1},
        {-1, 1,  3, -1, -1, -1},
        {-1, 4, -1, -1, -1, -1},
        {8,  4, -1, -1,  5, -1},
        {-1, 7, -1,  6, -1, -1},
        {-1, 7, -1, -1, -1, -1},
        {8,  7, -1, -1, -1, -1},
        {8, -1, -1, -1, -1, -1}
    };
    int opt(char c){
        if(c==' ') return 0;
        if(c>='0' && c<='9') return 1;
        if(c=='.') return 2;
        if(c=='+' || c=='-') return 3;
        if(c=='e' || c=='E') return 4;
        return 5;
    }
public:
    bool isNumber(string s) {
        int st = 0;
        s += " ";
        for(char c: s){
            st = stat[st][opt(c)];
            if(st==-1) return false;
        }
        return st==8;
    }
};
```

□ 之后重温一下 @Zebo L

249

1. 自己做下hash就行了

```
class Solution {
    #define MOVE(c, c0) char(int((c) - (c0) + 26)%26 + 'a')
public:
    vector<vector<string>> groupStrings(vector<string>& strings) {
        unordered_map<string, vector<string>> D;
        for(string s: strings){
            if(s.empty()) D[s].push_back(s);
            else{
                string tmp = s;
                for(char &c: tmp) c = MOVE(c, s[0]);
                D[tmp].push_back(s);
            }
        }
        vector<vector<string>> ans;
        for(auto p: D) ans.push_back(p.second);
        return ans;
    }
};
```