August 14, 2018 题目：569,282,314,400,841,496,66,627,598,412,131,393,694,379

569

SQL

282

1. 伤痛，一直做不对

□ Solve it!!!! @Zebo L

314

1. BFS, 次序问题好烦

```
class Solution {
    typedef vector<int> vi;
    typedef pair<TreeNode *, int> Ti;
    vector<vi> left, right;
public:
    vector<vector<int>> verticalOrder(TreeNode* root) {
        if(!root) return right;
        queue<Ti> Q;
        Q.push(Ti(root, 0));
        while(!Q.empty()){
            TreeNode *r = Q.front().first;
            int idx = Q.front().second;
            Q.pop();
            if(idx >= 0){
                if(right.size() < idx+1) right.push_back(vi
());
                right[idx].push_back(r->val);
            }
            else{
                if(left.size() < abs(idx)) left.push_back(vi());
                left[abs(idx)-1].push_back(r->val);
            }
            if(r->left) Q.push(Ti(r->left, idx-1));
            if(r->right) Q.push(Ti(r->right, idx+1));
        }
        vector<vi> ans(left.size()+right.size());
        reverse(left.begin(), left.end());
        copy(left.begin(), left.end(), ans.begin());
        copy(right.begin(), right.end(), ans.begin() + left.size());
        return ans;
    }
};
```

2. 用了DFS结果好像顺序不行,重新用BFS。TreeMap会增加complexity

```
class Solution {
    public List<List<Integer>> verticalOrder(TreeNode root) {
        List<List<Integer>> res = new ArrayList<>();
        if (root == null) return res;
        Map<Integer, List<Integer>> map = new TreeMap<>();
        Queue<TreeNode> q = new LinkedList<>();
        q.offer(root);
        Queue<Integer> level = new LinkedList<>();
        level.offer(0);
        while (!q.isEmpty()) {
            TreeNode node = q.poll();
            int l = level.poll();
            if (!map.containsKey(l)) {
                map.put(l, new ArrayList<>());
            }
            map.get(l).add(node.val);
            if (node.left != null) {
                q.offer(node.left);
                level.offer(l - 1);
            }
            if (node.right != null) {
                q.offer(node.right);
                level.offer(l + 1);
            }
        }

        res.addAll(map.values());
        return res;
    }
}
```

400

1. 注意细节就行了：

```
class Solution {
public:
    int findNthDigit(int n) {
        long K = 9, L = 1, M = 1;
        --n;
        while(n >= K*L){
            n -= K*L;
            K *= 10;
            ++L;
            M *= 10;
        }
        long res = M + n/L, dig = n%L;
        while(dig){
            res %= M;
            --dig;
            M/=10;
        }
        return res/M;
    }
};
```

841

1. 简化版dfs:

```
class Solution {
public:
    bool canVisitAllRooms(vector<vector<int>>& rooms) {
        unordered_set<int> S{0};
        queue<int> Q;
        Q.push(0);
        while(!Q.empty()){
            for(int j: rooms[Q.front()]) if(!S.count(j)){
                S.insert(j);
                Q.push(j);
            }
            Q.pop();
        }
        return S.size() == rooms.size();
    }
};
```

496

1. 倒装stack

```
class Solution {
public:
    vector<int> nextGreaterElement(vector<int>& A, vector<int>& B) {
        unordered_map<int, int> NG;
        stack<int> G;
        for(int i=B.size()-1; i>=0; --i){
            while(!G.empty() && G.top()<=B[i]) G.pop();
            NG[B[i]] = (G.empty()?-1:G.top());
            G.push(B[i]);
        }
        for(int &k: A) k = NG[k];
        return A;
    }
};
```

66

1. Straightforward:

```
class Solution {
public:
    vector<int> plusOne(vector<int>& digits) {
        vector<int> ans;
        for(int i=digits.size()-1, cur=1; i>=0||cur; --i){
            cur += (i>=0? digits[i]:0);
            ans.push_back(cur%10);
            cur /= 10;
        }
        reverse(ans.begin(), ans.end());
        return ans;
    }
};
```

627

又是SQL

598

1. Short is Beauty 系列。（开始看错题了，以为是矩阵中间某长方形内所有数字加一）

```python
class Solution(object):
    def maxCount(self, m, n, ops):
        return min([x for x, y in ops] + [m]) * min([y for x, y in ops] + [n])
```

412

1. Short is Beauty 系列：

```python
class Solution(object):
    def fizzBuzz(self, n):
        return [str(i)*bool(i%3 and i%5) + "Fizz"*(not i%3) + "Buzz"*(not i%5) for i in range(1, n+1)]
```

131

1. DFS: dfs 函数中把第二个argument 换成reference，runtime会从 `8s` 变到 `4s`

```cpp
class Solution {
    typedef vector<int> vi;
    typedef vector<string> vs;
    vector<unordered_set<int>> E;
    void dfs(vector<vs> &ans, vs cur, int i, string &s){
        if(i==s.size()){
            ans.push_back(cur);
            return;
        }
        for(auto j: E[i]) {
            cur.push_back(s.substr(i, j-i));
            dfs(ans, cur, j, s);
            cur.pop_back();
        }
    }
public:
    vector<vector<string>> partition(string s) {
        E.resize(s.size());
        for(int i=0; i<s.size(); ++i) E[i].insert(i+1);
        for(int l=2; l<=s.size(); ++l) for(int i=0; i+l<=s.size(); ++i) if(s[i]==s[i+l-1] && (l==2 || E[i+1].count(i+l-1))){
            E[i].insert(i+l);
        }
        vector<vs> ans;
        dfs(ans, vs(), 0, s);
        return ans;
    }
};
```

393

1. SB 题，文字游戏：

```
class Solution {
public:
    bool validUtf8(vector<int>& data) {
        if(data.empty()) return true;
        int n = 0;
        for(int x: data){
            if(!n){
                while(n<8 && 1&(x>>(7-n))) ++n;
                if(n==1 || n>4) return false;
                if(n) --n;
            }
            else{
                if(!(1&(x>>7)) || 1&(x>>6)) return false;
                --n;
            }
        }
        return !n;
    }
};
```

694

1. 不要耍小聪明: 用 `set<pair<int, int>>` 做hash

```
class Solution {
    typedef pair<int, int> ii;
    int d[4] = {1, -1, 0, 0};
public:
    int numDistinctIslands(vector<vector<int>>& grid) {
        if(grid.empty() || grid[0].empty()) return 0;
        int n = grid.size(), m = grid[0].size();
        set<set<ii>> I;
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j) if(grid[i][j]){
            queue<int> Q;
            Q.push(i*m + j);
            set<ii> tmp;
            grid[i][j] = 0;
            while(!Q.empty()){
                int x = Q.front()/m, y = Q.front()%m;
                Q.pop();
                for(int k=0; k<4; ++k){
                    int w = x + d[k], v = y + d[3-k];
                    if(w>=0 && w<n && v>=0 && v<m && grid[w]
[v]){
                        Q.push(w*m + v);
                        grid[w][v] = 0;
                        tmp.insert(ii(w-i, v-j));
                    }
                }
            }
            I.insert(tmp);
        }
        return I.size();
    }
};
```

379

1. 注意release 的时候check一下即可：

```
class PhoneDirectory {
    int cap, n;
    unordered_set<int> re;
public:
    PhoneDirectory(int maxNumbers): cap(maxNumbers), n(0) {}
    int get() {
        if(n==cap && re.empty()) return -1;
        if(!re.empty()){
            int k = *re.begin();
            re.erase(re.begin());
            return k;
        }
        ++n;
        return n-1;
    }
    bool check(int number) { return number>=n || re.count(number); }
    void release(int number) { if(!check(number)) re.insert(number); }
};
```