August 26, 2018 题目：753,279,760,847,702,384,553,711,185,636,671,458,156,18,130

753

1. 这个贪心怎么想也有点牵强附会：`32 ms`

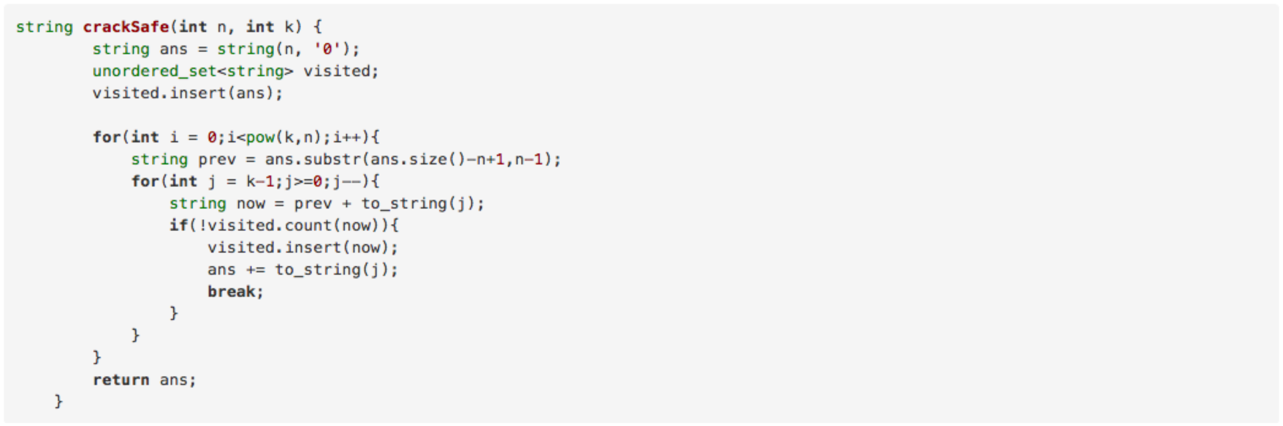
```
class Solution {
    int N, M;
    string res;
    bool dfs(int init, string ans, int&n, int&k, unordered_set
<int> &rest){
        if(rest.empty()) {
            res = ans;
            return true;
        }
        init = (init%M) * k;
        for(int j=0; j<k; ++j) if(rest.count(init + j)){
            ans.push_back(char(j+'0'));
            rest.erase(init+j);
            if(dfs(init + j, ans, n, k, rest)) return true;
            rest.insert(init+j);
            ans.pop_back();
        }
        return false;
    }
public:
    string crackSafe(int n, int k) {
        N = int(pow(k, n));
        M = int(pow(k, n-1));
        unordered_set<int> rest;
        for(int i=0; i<N; ++i) rest.insert(i);
        for(int j=0; j<N; ++j){
            string ans;
            int m = j;
            for(int i=0; i<n; ++i) {
                ans.push_back(char(m%k + '0'));
                m /= k;
            }
            reverse(ans.begin(), ans.end());
            rest.erase(j);
            if(dfs(j, ans, n, k, rest)) return res;
            rest.insert(j);
        }
        return "";
    }
};
```

2. 还能直接调用定理，我也是醉了
   https://en.wikipedia.org/wiki/De_Bruijn_sequence#Example_using_inverse_Burrows%E2%80%94Wheeler_transform
   - [ ] 再看看 @Zebo L

```
class Solution {
    // Use Burrows-Wheeler Transform to construct De Bruijn se
quence
public:
    string crackSafe(int n, int k) {
        int M = int(pow(k, n-1));
        string ans;
        vector<int> transform(k * M, 0), status(k * M, 1);
        for(int i=0; i<k * M; ++i) transform[i] = (i%M) * k +
i/M;
```

```
        for(int i=0; i<k * M; ++i){
            int current = i;
            while(status[current]){
                ans.push_back(char(current/M + '0'));
                status[current] = 0;
                current = transform[current];
            }
        }
        for(int i=0; i<n-1; ++i) ans.push_back(ans[i]);
        return ans;
    }
};
```

3. 直接greedy 居然也能过，这题的命题也太弱了：



```
string crackSafe(int n, int k) {
        string ans = string(n, '0');
        unordered_set<string> visited;
        visited.insert(ans);

        for(int i = 0;i<pow(k,n);i++){
            string prev = ans.substr(ans.size()-n+1,n-1);
            for(int j = k-1;j>=0;j--){
                string now = prev + to_string(j);
                if(!visited.count(now)){
                    visited.insert(now);
                    ans += to_string(j);
                    break;
                }
            }
        }
        return ans;
    }
```

279

1. dp:

```
class Solution {
public:
    int numSquares(int n) {
        vector<int> dp(n+1, n);
        dp[0] = 0;
        for(int j=0; j<n; ++j){
            for(int k=1; j<=n-k*k; ++k) dp[j + k*k] = min(dp[j
+ k*k], 1+dp[j]);
        }
        return dp[n];
    }
};
```

760

1. 难点何在：

```
class Solution {
public:
    vector<int> anagramMappings(vector<int>& A, vector<int>&
B) {
        unordered_map<int, queue<int>> Q;
        for(int i=0; i<B.size(); ++i) Q[B[i]].push(i);
        for(int &k: A){
            int j = Q[k].front();
            Q[k].pop();
            k = j;
        }
        return A;
    }
};
```

847

1. bitmask

```
class Solution {
    class Tuple {
        int val;
        int mask;
        int cost;
        public Tuple(int val, int mask, int cost) {
            this.val = val;
            this.mask = mask;
            this.cost = cost;
        }

        @Override
        public boolean equals(Object o) {
            if (o instanceof Tuple) {
                Tuple p = (Tuple) o;
                return this.val == p.val && this.mask == p.mas
k && this.cost == p.cost;
            }
            return false;
        }

        @Override
        public int hashCode() {
            return Objects.hash(val, mask, cost);
        }
    }
    public int shortestPathLength(int[][] graph) {
        int len = graph.length;
        Queue<Tuple> q = new LinkedList<>();
        Set<Tuple> set = new HashSet<>();
        for (int i = 0; i < len; i++) {
            int tmp = (1 << i);
            q.offer(new Tuple(i, tmp, 0));
            set.add(new Tuple(i, tmp, 0));
        }
        while (!q.isEmpty()) {
            Tuple cur = q.poll();
            if (cur.mask == ((1 << len) - 1)) return cur.cost;
            int[] neighbor = graph[cur.val];

            for (int v : neighbor) {
                int mask = cur.mask;
                mask = mask | (1 << v);
                Tuple t = new Tuple(v, mask, 0);
                if (!set.contains(t)) {
                    q.offer(new Tuple(v, mask, cur.cost + 1));
                    set.add(t);
                }
            }
        }
        return -1;
    }
}
```

2. BFS + memo, 不知道为什么DFS + memo一直做不对：

```
class Solution {
    typedef pair<int, int> ii;
    const int inf=1E9;
    int dp[14][5000];
public:
    int shortestPathLength(vector<vector<int>>& graph) {
        int n = graph.size(), M = (1<<(int)graph.size()) - 1;
        memset(dp, -1, sizeof(dp));
        graph.push_back(vector<int>());
        for(int i=0; i<n; ++i) graph[n].push_back(i);
        queue<ii> Q;
        Q.push(ii(n, 0));
        while(!Q.empty()){
            int i = Q.front().first, stat = Q.front().second;
            if(stat == M) return dp[i][stat];
            Q.pop();
            for(int j: graph[i]) if(dp[j][stat|(1<<j)] == -1)
{
                dp[j][stat | (1<<j)] = dp[i][stat] + 1;
                Q.push(ii(j, stat | (1<<j)));
            }
        }
        return -1;
    }
};
```

702

1. Bsearch:

```
class ArrayReader;
class Solution {
public:
    int search(const ArrayReader& reader, int target) {
        if(target > 10000 || target < reader.get(0)) return -
1;
        if(target == reader.get(0)) return 0;
        int l = 0, r = 2E4+1;
        while(l < r-1){
            int c = (l+r)/2;
            if(reader.get(c) < target) l = c;
            else r = c;
        }
        return (reader.get(r)==target? r: -1);
    }
};
```

384

1.

```
class Solution {
    int[] nums;
    Random r;
    public Solution(int[] nums) {
        this.nums = nums;
        r = new Random();
    }

    /** Resets the array to its original configuration and ret
urn it. */
    public int[] reset() {
        return nums;
    }

    /** Returns a random shuffling of the array. */
    public int[] shuffle() {
        int[] tmp = Arrays.copyOf(nums, nums.length);
        for (int i = tmp.length - 1; i >= 0; i--) {
            int index = r.nextInt(i + 1);
            int t = tmp[index];
            tmp[index] = tmp[i];
            tmp[i] = t;
        }
        return tmp;
    }
}
```

2. 小细节！！

```
class Solution {
    vector<int> origin, random;
public:
    Solution(vector<int> nums): origin(nums), random(nums){}
    vector<int> reset() { return this->origin; }
    vector<int> shuffle() {
        for(int i=origin.size()-1; i>0; --i) swap(random[i], random[rand()%(i+1)]);
        return random;
    }
};
```

553

1. 让结果最大的方式就是让第一个除号之后的值最小。那就是一直除后面的数。。

```
class Solution {
    public String optimalDivision(int[] nums) {
        if (nums == null || nums.length == 0) return "";
        StringBuilder sb = new StringBuilder();
        sb.append(nums[0]);
        if (nums.length == 1) return sb.toString();
        if (nums.length == 2) {
            sb.append("/").append(nums[1]);
            return sb.toString();
        }
        sb.append("/(");
        for (int i = 1; i < nums.length; i++) {
            sb.append(nums[i]);
            if (i == nums.length - 1) sb.append(")");
            else sb.append("/");
        }
        return sb.toString();
    }
}
```

2. Short is Beauty 系列：

```
class Solution(object):
    def optimalDivision(self, nums):
        return "/".join([str(x) for x in nums]) if len(nums)<=
2 else str(nums[0])+"/("+"/".join([str(x) for x in nums[1:]])
+")"
```

711

1. 直接用矩阵做hash：

```
class Solution {
    typedef vector<int> vi;
    typedef pair<int, int> ii;
    vector<vi> refl(vector<vi> A){
        for(auto &vec: A) reverse(vec.begin(), vec.end());
        return A;
    }
    vector<vi> rot(vector<vi> A){
        int n = A.size(), m = A[0].size();
        vector<vi> B(m, vi(n, 0));
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j) B[j][n-i
-1] = A[i][j];
        return B;
    }
    int d[4] = {1, -1, 0, 0};
public:
    int numDistinctIslands2(vector<vector<int>>& G) {
        if(G.empty() || G[0].empty()) return 0;
        int n = G.size(), m = G[0].size();
        set<vector<vi>> Pool;
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j) if(G[i][j]){
            queue<ii> Q;
            set<ii> S{ii(i, j)};
            G[i][j] = 0;
            int i0 = i, i1 = i+1, j0 = j, j1 = j+1;
            Q.push(ii(i, j));
            while(!Q.empty()){
                int x = Q.front().first, y = Q.front().second;
                i0 = min(i0, x);
                j0 = min(j0, y);
                i1 = max(i1, x+1);
                j1 = max(j1, y+1);
                Q.pop();
                for(int k=0; k<4; ++k){
                    int x1 = x + d[k], y1 = y + d[3-k];
                    if(x1>=0 && x1<n && y1>=0 && y1 <m && G[x1][y1] && !S.count(ii(x1, y1))){
                        G[x1][y1] = 0;
                        S.insert(ii(x1, y1));
                        Q.push(ii(x1, y1));
                    }
                }
            }
            vector<vi> A(i1-i0, vi(j1-j0, 0));
            for(auto p: S){
                A[p.first-i0][p.second-j0] = 1;
            }
            int cnt = 4;
            vector<vector<vi>> tmp;
            do{
                tmp.push_back(A);
                tmp.push_back(refl(A));
                A = rot(A);
                --cnt;
            }while(cnt);
            sort(tmp.begin(), tmp.end());
            Pool.insert(tmp[0]);
        }
        return (int)Pool.size();
    }
};
```

185

SQL

636

1. stack is used to maintain the id

```
class Solution {
    public int[] exclusiveTime(int n, List<String> logs) {
        int[] res = new int[n];
        int pre = 0;
        Stack<Integer> stack = new Stack<>();
        for (String log : logs) {
            String[] l = log.split(":");
            if (!stack.isEmpty()) res[stack.peek()] += Intege
r.parseInt(l[2]) - pre;
            pre = Integer.parseInt(l[2]);
            if (l[1].equals("start")) {
                stack.push(Integer.parseInt(l[0]));
            } else {
                res[stack.pop()]++;
                pre++;
            }
        }
        return res;
    }
}
```
2. 方法同上，注意区间是`inclusively`定义的：
```
class Solution {
    typedef vector<int> vi;
public:
    vector<int> exclusiveTime(int n, vector<string>& logs) {
        vector<vi> events;
        for(string s: logs){
            int id = stoi(s);
            s = s.substr(s.find(':') + 1);
            int stat = 0;
            if(s[0] == 's') stat = 1;
            s = s.substr(s.find(':') + 1);
            int tm = stoi(s);
            if(!stat) ++tm;
            events.push_back({tm, id, stat});
        }
        stack<int> S;
        vector<int> ans(n, 0);
        for(int i=0; i<events.size(); ++i){
            if(!S.empty()){
                ans[S.top()] += events[i][0] - events[i-1][0];
            }
            if(events[i][2]) S.push(events[i][1]);
            else S.pop();
        }
        return ans;
    }
};
```

671

1.审题

```
class Solution {
    public int findSecondMinimumValue(TreeNode root) {
        if (root == null) return -1;
        if (root.left == null && root.right == null) return -
1;
        int left = root.left.val;
        int right = root.right.val;

        if (left == root.val) left = findSecondMinimumValue(ro
ot.left);
        if (right == root.val) right = findSecondMinimumValue
(root.right);
        if (left != -1 && right != -1) {
            return Math.min(left, right);
        } else if (left != -1) {
            return left;
        } else {
            return right;
        }
    }
}
```

2. 正常preorder 即可：

```
class Solution {
public:
    int findSecondMinimumValue(TreeNode* root) {
        long a = long(INT_MAX)+1, b = long(INT_MAX) + 1;
        stack<TreeNode *> S;
        while(root || !S.empty()){
            while(root){
                if(root->val<a) {
                    b = a;
                    a = root->val;
                }
                else if(root->val > a && root->val < b) b = ro
ot->val;
                if(root->right) S.push(root->right);
                root = root->left;
            }
            if(!S.empty()){
                root = S.top();
                S.pop();
            }
        }
        return (b>long(INT_MAX)? -1: b);
    }
};
```

458

1. 我擦，看不懂题啊
    - SB啊：https://leetcode.com/problems/poor-pigs/discuss/94266/Another-explanation-and-solution ，理解题意是 hard++ 难度。

```
class Solution {
public:
    int poorPigs(int buckets, int a, int b) {
        return floor(log(buckets)/log(int(b/a)+1) - 0.000001)
+ 1;
    }
};
```

156

1. recursion

```
class Solution {
    public TreeNode upsideDownBinaryTree(TreeNode root) {
        if (root == null) return root;
        if (root.left == null && root.right == null) return ro
ot;
        TreeNode newRoot = upsideDownBinaryTree(root.left);
        root.left.left = root.right;
        root.left.right = root;
        root.left = null;
        root.right = null;
        return newRoot;
    }
}
```

2. 之前用的recursion，突然发现`stack`也行：

```
class Solution {
public:
    TreeNode* upsideDownBinaryTree(TreeNode* root) {
        TreeNode lead(0);
        TreeNode *pre = &lead;
        stack<TreeNode *> S;
        while(root) {
            S.push(root);
            root = root->left;
        }
        while(!S.empty()){
            pre->right = S.top();
            pre = S.top();
            S.pop();
            if(!S.empty()){
                pre->left = S.top()->right;
            }
            else pre->left = pre->right = NULL;
        }
        return lead.right;
    }
};
```

18

1. $O(n^2)$ + Two sum

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

- [ ] 这个值得一看：https://leetcode.com/problems/4sum/discuss/163559/C++-solution-for-all-Ksum-question

130

1. Frontier + BFS: 也有dfs解法

```
class Solution {
    int d[4] = {1, -1, 0, 0};
public:
    void solve(vector<vector<char>>& B) {
        if(B.size() <= 2 || B[0].size() <= 2) return;
        int n = B.size(), m = B[0].size();
        queue<int>Q;
        unordered_set<int> rest;
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j) if(B[i]
[j] == 'O'){
            if(!i || i==n-1 || !j || j==m-1) Q.push(i*m + j);
            else rest.insert(i*m + j);
        }
        while(!Q.empty()){
            int i = Q.front()/m, j = Q.front()%m;
            Q.pop();
            for(int k=0; k<4; ++k){
                int x = i+d[k], y = j + d[3-k];
                if(rest.count(x * m + y)){
                    Q.push(x * m + y);
                    rest.erase(x * m + y);
                }
            }
        }
        for(int k: rest) B[k/m][k%m] = 'X';
    }
};
```

2. DFS

```
class Solution {
    public void solve(char[][] board) {
        if (board == null || board.length == 0) return;
        int M = board.length, N = board[0].length;
        for (int i = 0; i < M; i++) {
            if (board[i][0] == 'O') {
                helper(board, i, 0);
            }
            if (board[i][N - 1] == 'O') {
                helper(board, i, N - 1);
            }
        }

        for (int i = 0; i < N; i++) {
            if (board[0][i] == 'O') {
                helper(board, 0, i);
            }
            if (board[M - 1][i] == 'O') {
                helper(board, M - 1, i);
            }
        }

        for (int i = 0; i < M; i++) {
            for (int j = 0; j < N; j++) {
                if (board[i][j] == 'O') {
                    board[i][j] = 'X';
                } else if (board[i][j] == '#') {
                    board[i][j] = 'O';
                }
            }
        }
    }
    public void helper(char[][] board, int x, int y) {
        if (x < 0 || y < 0 || x >= board.length || y >= board
[0].length || board[x][y] != 'O') return;
        board[x][y] = '#';
        helper(board, x + 1, y);
        helper(board, x - 1, y);
        helper(board, x, y + 1);
        helper(board, x, y - 1);
    }
}
```