August 5, 2018 题目：348,527,298,481,768,94,488,120,44,46,226,112,71

348

1. O(1) 的奇技淫巧, count row, count col, count diagonal and anti-diagonal, return 3 states, 0 for no one is winning, -1, 1 for each player
2. 同上

```
class TicTacToe {
public:
    typedef vector<int> vi;
    typedef vector<bool> vb;
    int n;
    vector<vi> P;
    vector<vb> G;
    TicTacToe(int t): P(vector<vi>(2, vi(2*n + 2, 0))), G(vector<vb>(n, vb(n, false))), n(t) {}
    int move(int row, int col, int player) {
        assert(!G[row][col]);
        G[row][col] = true;
        P[player-1][row]++;
        P[player-1][n + col]++;
        if(row == col) P[player-1][2*n]++;
        if(row == n-1-col) P[player-1][2*n+1]++;
        if(P[player-1][n+col]>=n || P[player-1][row]>=n || P[player-1][2*n]>=n || P[player-1][2*n+1]>=n) return player;
        return 0;
    }
};
```

527

1. 之前把题意理解为：把原`dict`中最少的`string` 变成缩写。。。

```
class Solution {
public:
    vector<string> wordsAbbreviation(vector<string>& dict) {
        set<string> S(dict.begin(), dict.end());
        for(int i=0; i<dict.size(); ++i) if(dict[i].size() > 3){
            string s = dict[i];
            int l = s.size(), res = 1;
            for(auto it=S.lower_bound(string(1, s[0])); it!=S.end()&& (*it)[0]==s[0]; ++it) if(*it!=s && (*it).size() == l && (*it)[l-1] == s[l-1]){
                int j=0;
                while(j<l && (*it)[j]==s[j]) ++j;
                res = max(res, j+1);
            }
            if(to_string(l-res-1).size() + res + 1 < l) dict[i] = s.substr(0, res) + to_string(l-res-1) + s.substr(l-1);
        }
        return dict;
    }
};
```

298

1. dfs

```
class Solution {
    int res;
    void dfs(TreeNode *root, int l){
        if(!root) return;
        res = max(res, l);
        if(root->left && root->left->val == root->val+1) dfs(root->left, l+1);
        else dfs(root->left, 1);
        if(root->right && root->right->val == root->val+1) dfs(root->right, l+1);
        else dfs(root->right, 1);
    }
public:
    int longestConsecutive(TreeNode* root) {
        res = 0;
        dfs(root, 1);
        return res;
    }
};
```

2. 把 val + 1 在dfs传下去的奇技淫巧

481

1. Concatenating char is much faster than concatenating string:

```
class Solution {
public:
    int magicalString(int n) {
        if(!n) return 0;
        string s = "122";
        int i = 2, cnt = 1, cur = 2;
        while(s.size()<n){
            cur = 3 - cur;
            for(int j=0; j<int(s[i]-'0')&&s.size()<n; ++j){
                s += char(cur + '0');
                if(cur==1) ++cnt;
            }
            ++i;
        }
        return cnt;
    }
};
```

768

1. Why hard?

```
class Solution {
public:
    int maxChunksToSorted(vector<int>& arr) {
        vector<int> M(arr);
        for(int i=1; i<arr.size(); ++i) M[i] = max(M[i], M[i-
1]);
        int ans = 1;
        for(int i=arr.size()-1, m=arr[arr.size()-1]; i; --i){
            m = min(m, arr[i]);
            if(M[i-1]<=m) ++ans;
        }
        return ans;
    }
};
```

2. hard to come up with such an idea lol

94

1. Practise hand

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

488

1. 一直以为是dfs迭代过多导致超时，最后才发现是个bug导致死循环。这种级别的题做了我一个多小时

```
class Solution {
    typedef vector<int> vi;
    typedef pair<string, vi> svi;
    const int inf = 1E8;
    string reduce(string s, int i){
        int j = i, k = i;
        while(j>=0 && s[j]==s[i]) --j;
        while(k<s.size() && s[k]==s[i]) ++k;
        while(j>=0 && k<s.size() && s[j]==s[k] && ((j && s[j]=
=s[j-1]) || (k<s.size()-1 && s[k]==s[k+1]))){
            char c = s[j];
            while(j>=0 && s[j]==c) --j;
            while(k<s.size() && s[k]==c) ++k;
        }
        return s.substr(0, j+1) + s.substr(k);
    }
    int getIdx(char c){
        switch(c){
            case 'R': return 0;
            case 'Y': return 1;
            case 'B': return 2;
            case 'G': return 3;
            case 'W': return 4;
        }
        return -1;
    }
    //map<svi, int> dp;
    int dfs(string s, vi v){
        if(s.empty()) return 0;
        //if(dp.count(svi(s, v))) return dp[svi(s, v)];
        //dp[svi(s, v)] = inf;
        int ans = inf;
        int cnt = 1, n = s.size();
        for(int i=1; i<n; ++i){
            if(s[i]==s[i-1]) ++cnt;
            else{
                if(cnt + v[getIdx(s[i-1])] >= 3){
                    v[getIdx(s[i-1])] += cnt - 3;
                    //dp[svi(s, v)] = min(dp[svi(s, v)], 3 - c
nt + dfs(reduce(s, i-1), v));
                    ans = min(ans, 3-cnt + dfs(reduce(s, i-1),
v));
                    v[getIdx(s[i-1])] += 3 - cnt;
                }
                cnt = 1;
            }
        }
        if(cnt + v[getIdx(s[n-1])] >= 3){
            v[getIdx(s[n-1])] += cnt - 3;
            //dp[svi(s, v)] = min(dp[svi(s, v)], 3 - cnt + dfs
(reduce(s, n-1), v));
            ans = min(ans, 3-cnt + dfs(reduce(s, n-1), v));
            v[getIdx(s[n-1])] += 3 - cnt;
        }
        //return dp[svi(s, v)];
        return ans;
    }
public:
    int findMinStep(string board, string hand) {
        vi init(5, 0);
        for(char c: hand) init[getIdx(c)]++;
        int ans = dfs(board, init);
        if(ans == inf) return -1;
        return ans;
    }
};
```

120

1. `O(1) space.`

```
class Solution {
public:
    int minimumTotal(vector<vector<int>>& T) {
        if(T.empty()) return 0;
        for(int i=T.size()-2; i>=0; --i){
            for(int j=0; j<=i; ++j) T[i][j] += min(T[i+1][j], 
T[i+1][j+1]);
        }
        return T[0][0];
    }
};
```

44

1. 刚做过：

```
class Solution {
public:
    bool isMatch(string s, string p) {
        int n = s.size(), m = p.size();
        vector<vector<bool>> dp(n+1, vector<bool>(m+1, fals
e));
        dp[n][m] = true;
        for(int j=m-1; j>=0 && p[j]=='*'; --j) dp[n][j] = tru
e;
        for(int i=n-1; i>=0; --i) for(int j=m-1; j>=0; --j){
            if(p[j] == '?') dp[i][j] = dp[i+1][j+1];
            else if(p[j] == '*'){
                for(int k=i; k<=n && !dp[i][j]; ++k) dp[i][j]
= dp[k][j+1];
            }
            else dp[i][j] = (s[i]==p[j]) && dp[i+1][j+1];
        }
        return dp[0][0];
    }
};
```

46

1. `n^n` dfs

```
class Solution {
    typedef vector<int> vi;
    typedef vector<bool> vb;
    void dfs(vector<vi>& ans, vi cur, vi& A, vb &mark){
        if(cur.size()==A.size()){
            ans.push_back(cur);
            return;
        }
        for(int i=0; i<A.size(); ++i) if(!mark[i]){
            cur.push_back(A[i]);
            mark[i] = true;
            dfs(ans, cur, A, mark);
            mark[i] = false;
            cur.pop_back();
        }
    }
public:
    vector<vector<int>> permute(vector<int>& nums) {
        vector<vi> ans;
        vb mark(nums.size(), false);
        dfs(ans, vi(), nums, mark);
        return ans;
    }
};
```

226

1. Recursion

```
class Solution {
public:
    TreeNode* invertTree(TreeNode* root) {
        if(!root) return NULL;
        invertTree(root->left);
        invertTree(root->right);
        swap(root->left, root->right);
        return root;
    }
};
```

112

1. Easy recursion

```
class Solution {
public:
    bool hasPathSum(TreeNode* root, int sum) {
        if(!root) return false;
        if(!root->left && !root->right) return sum == root->va
l;
        return hasPathSum(root->left, sum-root->val) || hasPat
hSum(root->right, sum-root->val);
    }
};
```

71

1. Python 好写太多了

```
class Solution(object):
    def simplifyPath(self, path):
        ans = []
        ref = [s for s in path.split('/') if s]
        for s in ref:
            if s == '..':
                if ans:
                    ans.pop();
            elif s != '.':
                ans.append(s)
        return '/' + '/'.join(ans)
```