September 3, 2018 题目：299,253,778,251,323,827,282,418,6,453,775,338,78,546,551,417

299

1. 按照题意直接写即可：

```
class Solution {
public:
    string getHint(string secret, string guess) {
        int a=0, b=0;
        vector<int> A(10, 0), B(10, 0);
        for(int i=0; i<secret.size(); ++i){
            if(secret[i] == guess[i]) ++a;
            else{
                ++A[int(secret[i]-'0')];
                ++B[int(guess[i]-'0')];
            }
        }
        for(int i=0; i<10; ++i) b += min(A[i], B[i]);
        return to_string(a) + "A" + to_string(b) + "B";
    }
};
```

253

1. Interval 迭代:

```
class Solution {
public:
    int minMeetingRooms(vector<Interval>& intervals) {
        map<int, int> range;
```

```
        for(auto i: intervals) {
            ++range[i.start];
            --range[i.end];
        }
        int ans = 0, tmp = 0;
        for(auto p: range){
            tmp += p.second;
            ans = max(ans, tmp);
        }
        return ans;
    }
};
```

778

1. DFS

```
class Solution {
    int time = 0;
    public int swimInWater(int[][] grid) {
        boolean[][] visited = new boolean[grid.length][grid
[0].length];
        int time = 0;

        while (!visited[grid.length - 1][grid[0].length - 1])
{
            visited = new boolean[grid.length][grid[0].lengt
h];
            helper(grid, 0, 0, time, visited);
            time++;
        }
        return time - 1;
    }
```

```
    private void helper(int[][] grid, int row, int col, int ti
me, boolean[][] visited) {
        if (row < 0 || row >= grid.length || col < 0 || col >=
grid[0].length || visited[row][col] || time < grid[row][col])
{
            return;
        }
        visited[row][col] = true;
        helper(grid, row + 1, col, time, visited);
        helper(grid, row - 1, col, time, visited);
        helper(grid, row, col + 1, time, visited);
        helper(grid, row, col - 1, time, visited);
    }
}
```

2. 典型的`Union Find`:

```
class Solution {
    typedef vector<int> vi;
    vi P;
    int getRoot(int j){
        if(P[j] == j) return j;
        return P[j] = getRoot(P[j]);
    }
    int d[4] = {1, -1, 0, 0};
    void rise(int q, const int&n, const int&m){
        P[q] = q;
        int i = q/m, j = q%m;
        for(int k=0; k<4; ++k){
            int x = i + d[k], y = j + d[3-k];
            if(x>=0 && x<n && y>=0 && y<m && P[x*m + y] >= 0){
                P[getRoot(q)] = getRoot(x*m + y);
            }
```

```
        }
    }
public:
    int swimInWater(vector<vector<int>>& grid) {
        int n = grid.size(), m = grid[0].size(), k = 0;
        P = vi(n*m, -1);
        map<int, vi> H;
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j) H[grid
[i][j]].push_back(i*m + j);
        for(auto p: H){
            for(int q: p.second) rise(q, n, m);
            if(P[0]>=0 && P[n*m-1]>=0 && getRoot(0) == getRoot
(n*m - 1)) return p.first;
        }
        return -1;
    }
};
```

251

1. queue :

```
class Vector2D {
    queue<int> Q;
public:
    Vector2D(vector<vector<int>>& vec2d) { for(auto vec: vec2
d) for(auto k: vec) Q.push(k); }
    int next() {
        int ans = Q.front();
        Q.pop();
        return ans;
    }
    bool hasNext() { return !Q.empty(); }
```

```
};
```

323

1. BFS, Union Find 均可：

```
class Solution {
    vector<int> P;
    int findRoot(int i){
        if(P[i] == i) return i;
        return P[i] = findRoot(P[i]);
    }
public:
    int countComponents(int n, vector<pair<int, int>>& edges)
{
        P.resize(n);
        for(int i=0; i<n; ++i) P[i]=i;
        for(auto p: edges) if(findRoot(p.first) != findRoot(p.
second)) P[findRoot(p.second)] = findRoot(p.first);
        int ans = 0;
        for(int i=0; i<n; ++i) ans += P[i] == i;
        return ans;
    }
};
```

827

1. DFS

```
class Solution {
    public int largestIsland(int[][] grid) {
        int m = grid.length, n = grid[0].length;
        int res = -1;
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
```

```
                if (grid[i][j] == 0) {
                    grid[i][j] = 1;
                    res = Math.max(res, helper(i, j, grid, new
boolean[m][n], m, n));
                    grid[i][j] = 0;
                }
            }
        }
        return res == -1 ? m * n : res;
    }

    public int helper(int i, int j, int[][] grid, boolean[][]
visited, int m, int n) {
        if (i < 0 || i >= m || j < 0 || j >= n || visited[i]
[j] || grid[i][j] != 1) {
            return 0;
        }
        visited[i][j] = true;
        return 1 + helper(i + 1, j, grid, visited, m, n) + hel
per(i - 1, j, grid, visited, m, n) + helper(i, j + 1, grid, vi
sited, m, n) + helper(i, j - 1, grid, visited, m, n);
    }
}
```

2. `Union Find`:

```
class Solution {
    vector<int> P, A;
    int findRoot(int j){
        if(P[j] == j) return j;
        return P[j] = findRoot(P[j]);
    }
    int d[4] = {1, -1, 0, 0};
```

```
    void merge(int q, const int&n, const int&m){
        P[q] = q;
        A[q] = 1;
        int i = q/m, j = q%m;
        for(int k=0; k<4; ++k){
            int x = i + d[k], y = j + d[3-k];
            if(x>=0 && x<n && y>=0 && y<n && P[x*m + y] >=0){
                int p = x*m + y;
                if(findRoot(q) != findRoot(p)){
                    A[findRoot(p)] += A[findRoot(q)];
                    P[findRoot(q)] = findRoot(p);
                }
            }
        }
    }
    int getArea(int q, const int&n, const int&m){
        int i = q/m, j= q%m, area = 1;
        set<int> I;
        for(int k=0; k<4; ++k){
            int x = i + d[k], y = j + d[3-k];
            if(x>=0 && x<n && y>=0 && y<n && P[x*m + y] >=0){
                int r = findRoot(x*m + y);
                if(!I.count(r)){
                    I.insert(r);
                    area += A[r];
                }
            }
        }
        return area;
    }
```

```
public:
    int largestIsland(vector<vector<int>>& grid) {
        int n = grid.size(), m = grid[0].size(), ans = 0;
        A = vector<int> (n*m, 0);
        P = vector<int> (n*m, -1);
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j) if(grid
[i][j]) merge(i*m+j, n, m);
        if(P[0] >= 0) ans = max(ans, A[findRoot(0)]);
        for(int i=0; i<n; ++i) for(int j=0; j<m; ++j) if(!grid
[i][j]) ans = max(ans, getArea(i*m + j, n, m));
        return ans;
    }
};
```

282

1. 想了一个月：结果还是慢到爆

```
class Solution {
    #define CI(c) long((c) - '0')
    void dfs(vector<string>&ans, string cur, vector<long> inf
o, const string&s){
        /*
        info[0]: target (target value)
        info[1]: state (state)
        info[2]: index (index in the string)
        info[3]: rslt (current calculated results)
        info[4]: prod (deal with products)
        info[5]: pres (present number)
        */
        long target = info[0], state = info[1], index = info
[2];
        long rslt = info[3], prod = info[4], pres = info[5];
```

```
        if(index == s.size()){
            if(state < 2 && rslt + prod*pres == target) ans.push_back(cur);
            return;
        }
        if(state < 2){
            dfs(ans, cur+'+', {target, 2, index, rslt+prod*pres, 1, 0}, s);
            dfs(ans, cur+'-', {target, 2, index, rslt+prod*pres, -1, 0}, s);
            dfs(ans, cur+'*', {target, 2, index, rslt, prod*pres, 0}, s);
        }
        if(state == 1) dfs(ans, cur+s[index], {target, 1, index+1, rslt, prod, pres*10+CI(s[index])}, s);
        if(state == 2) dfs(ans, cur+s[index], {target, long(s[index]!='0'), index+1, rslt, prod, CI(s[index])}, s);
    }
public:
    vector<string> addOperators(string num, int target) {
        long n = num.size();
        vector<string> ans;
        dfs(ans, "", {long(target), 2, 0, 0, 1, 0}, num);
        return ans;
    }
};
```

- [ ] Need Look at Other People's solution @Zebo L

418

1. 利用状态transition：

- [ ] 记住这个解法，好不容易想出来的 @Zebo L

```
class Solution {
```

```
    typedef pair<int, int> ii;
public:
    int wordsTyping(vector<string>& sentence, int rows, int co
ls) {
        int n = sentence.size();
        ++cols;
        vector<int> P(n+1, 0);
        for(int i=1; i<=n; ++i) P[i] = P[i-1] + 1 + sentence[i
-1].size();
        vector<ii> V(n, ii(0, -1));
        for(int i=0; i<n; ++i){
            int k = i, m = cols;
            if(m >= P[n] - P[k]){
                m -= P[n] - P[k];
                ++V[i].first;
                k = 0;
            }
            V[i].first += m / P[n];
            m %= P[n];
            int l = k, r = n;
            while(l < r-1){
                int c = (l + r)/2;
                if(P[c] - P[k]<=m) l = c;
                else r = c;
            }
            V[i].second = l;
        }
        int start = 0, ans = 0;
        for(int i=0; i<rows; ++i){
            ans += V[start].first;
```

```
            start = V[start].second;
        }
        return ans;
    }
};
```

6

1. 细心点就行了：

```
class Solution {
public:
    string convert(string s, int numRows) {
        int n = s.size(), k = numRows* 2-2;
        if(!k) return s;
        string ans;
        for(int i=0; i<(n+k-1)/k; ++i) ans += s[i*k];
        for(int j=1; j<numRows-1; ++j) for(int i=0; i<(n+k-1)/
k; ++i){
            if(i*k + j < n) ans += s[i*k + j];
            if(i*k + k - j < n) ans += s[i*k + k - j];
        }
        for(int i=0; i<(n+k-1)/k; ++i) if(i*k + numRows-1 < n)
ans += s[i*k + numRows-1];
        return ans;
    }
};
```

453

1. Short is Beauty 系列：

```
class Solution(object):
    def minMoves(self, nums):
        return sum(nums) - len(nums) * min(nums)
```

775

1. Semi Short is Beauty 系列：

```
class Solution {
public:
    bool isIdealPermutation(vector<int>& A) {
        for(int j=A.size()-1, m=A[j]; j>1; --j){
            m = min(m, A[j]);
            if(m <= A[j-2]) return false;
        }
        return true;
    }
};
```

338

1. Semi Short is Beauty 系列：

```
class Solution {
public:
    vector<int> countBits(int num) {
        vector<int> ans(num+1, 0);
        for(int i=1; i<=num; ++i) ans[i] = (i%2?ans[i-1]+1: an
s[i/2]);
        return ans;
    }
};
```

78

1. Bit Manipulation:

```
class Solution {
public:
    vector<vector<int>> subsets(vector<int>& nums) {
        int n = nums.size();
```

```
        vector<vector<int>> ans;
        for(int k=0; k<(1<<n); ++k){
            vector<int> cur;
            for(int j=0; j<n; ++j) if(1&(k>>j)) cur.push_back
(nums[j]);
            ans.push_back(cur);
        }
        return ans;
    }
};
```

546

551

1. Too easy for you to meet in a interview:

```
class Solution {
public:
    bool checkRecord(string s) {
        int cntA = 0, cntL = 0;
        for(char c: s){
            if(c == 'L'){
                ++cntL;
                if(cntL > 2) return false;
            }
            else{
                cntL = 0;
                if(c == 'A'){
                    ++cntA;
                    if(cntA > 1) return false;
                }
            }
```

```
        }
        return true;
    }
};
```

417

1. 个人prefer BFS，虽然dfs + memo 也能做：

```
class Solution {
    typedef pair<int, int> ii;
    int d[4] = {1, -1, 0, 0}, n, m;
    void bfs(unordered_set<int> &S, vector<vector<int>>& M){
        queue<int> Q;
        for(int k: S) Q.push(k);
        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(x>=0 && x<n && y>=0 && y<m && !S.count(x*m+
y) && M[x][y]>=M[i][j]){
                    S.insert(x*m + y);
                    Q.push(x*m + y);
                }
            }
        }
    }
public:
    vector<pair<int, int>> pacificAtlantic(vector<vector<int>>
& matrix) {
        vector<pair<int, int>> ans;
        if(matrix.empty() || matrix[0].empty()) return ans;
```

```
        n = matrix.size();
        m = matrix[0].size();
        unordered_set<int> P, A;
        for(int i=0; i<n; ++i){
            P.insert(i*m + 0);
            A.insert(i*m + m - 1);
        }
        for(int j=0; j<m; ++j){
            P.insert(0 + j);
            A.insert((n-1)*m + j);
        }
        bfs(P, matrix);
        bfs(A, matrix);
        for(int k: P) if(A.count(k)) ans.push_back(ii(k/m, k%
m));
        return ans;
    }
};
```