June 30, 2018

@Mingze X @Chong T @Zhaorui D @Yu S

577 (MySql)

438

goodspeed solution:

idea: use int[26] to hash p:

code:

```
class Solution {
public:
    vector<int> findAnagrams(string s, string p) {
        vector<int> ans, ref(26, 0), cnt(26, 0);
        if(p.size() > s.size()) return ans;
        for(char c: p) ref[int(c-'a')] += 1;
        for(int j=0; j<p.size(); ++j) cnt[int(s[j]-'a')] += 1;
        int start = 0, end = p.size();
        while(true){
            if(cnt == ref) ans.push_back(start);
            if(end >= s.size()) break;
            cnt[int(s[end++]-'a')] += 1;
            cnt[int(s[start++]-'a')] -= 1;
        }
        return ans;
    }
};
```

Mingze's solution:

lang: JS

code:

```
/**
```

```
 * @param {string} s
 * @param {string} p
 * @return {number[]}
 */
var findAnagrams = function(s, p) {
    let ans = [], dicS = {}, dicP = {};

    if (p.length > s.length) return ans;

    for (let i = 'a'.charCodeAt(); i <= 'z'.charCodeAt(); i++)
{
        dicS[String.fromCharCode(i)] = 0;
        dicP[String.fromCharCode(i)] = 0;
    }

    p.split('').map(c => { dicP[c] += 1 });

    for (let i = 0; i < p.length; i++) {
        dicS[s[i]] += 1;
    }

    let start = 0, end = p.length;
    while (end <= s.length) {
        if (JSON.stringify(dicS) == JSON.stringify(dicP)) ans.
push(start);
        dicS[s[start++]] -= 1;
        dicS[s[end++]] += 1;
    }

    return ans;
```

```
};
```

840

goodspeed solution:

idea: check magic one by one

code:

```
class Solution {
    bool isMagic(int i, int j, vector<vector<int>>& grid){
        int ref = 0;
        vector<int> sums(8, 0);
        for(int di=0;di<3;++di) {
            for(int dj=0;dj<3;++dj){
                int val = grid[i+di][j+dj];
                if(val > 9 || val < 1) return false;
                if(1&(ref>>val)) return false;
                ref |= (1<<val);
                sums[di] += val;
                sums[3 + dj] += val;
                if(di==dj) sums[6] += val;
                if(di+dj == 2) sums[7] += val;
            }
        }
        for(int m: sums) if(m!=15) return false;
        return true;
    }
public:
    int numMagicSquaresInside(vector<vector<int>>& grid) {
        if(grid.empty() || grid[0].empty()) return 0;
        int ans = 0;
        for(int i=0; i<(int)grid.size()-2 ; ++i) for(int j=0;
j<(int)grid[0].size()-2; ++j){
```

```
            if(isMagic(i, j, grid)) ans += 1;
        }
        return ans;
    }
};
```

Mingze's solution:

lang: JS

code:

```
/**
 * @param {number[][]} grid
 * @return {number}
 */
var numMagicSquaresInside = function(grid) {
    let isMagic = (row, col) => {
        let dict = {}, rowSum = [0, 0, 0], colSum = [0, 0, 0],
xSum = [0, 0];
        for (let i = 0; i < 3; i++) {
            for (let j = 0; j < 3; j++) {
                let val = grid[row+i][col+j];
                if (val < 1 || val > 9) return false;
                if (dict[val]) return false;
                dict[val] = true;
                rowSum[i] += val;
                colSum[j] += val;
                if (i === j) xSum[0] += val;
                if (i + j === 2) xSum[1] += val;
            }
        }
        for (let s of rowSum) if (s !== 15) return false;
        for (let s of colSum) if (s !== 15) return false;
```

```
        for (let s of xSum) if (s !== 15) return false;
        return true;
    }

    let ans = 0;
    for (let i = 0; i < grid.length - 2; i++) {
        for (let j = 0; j < grid[0].length - 2; j++) {
            if (isMagic(i, j)) ans++;
        }
    }

    return ans;
};
```

254

goodspeed solution:

idea: dfs

code:

```
class Solution {
    void genFacts(int mini, int res, vector<int> cur, vector<v
ector<int>> & ans){
        for(int j=mini; j<=sqrt(res); ++j)  if(res%j==0 && re
s/j>=j) {
            cur.push_back(j);
            genFacts(j, res/j, cur, ans);
            cur.pop_back();
        }
        cur.push_back(res);
        if(cur.size()>1) ans.push_back(cur);
    }
```

```
public:
    vector<vector<int>> getFactors(int n) {
        vector<vector<int>> ans;
        genFacts(2, n, vector<int>(), ans);
        return ans;
    }
};
```

263

Mingze's solution:

lang: Python

code:

```
class Solution(object):
    def isUgly(self, num):
        """
        :type num: int
        :rtype: bool
        """
        if num <= 0: return False
        for i in [2, 3, 5]:
                while num % i == 0:
                        num /= i
        return num == 1
```

goodspeed's solution:

lang: c++

idea: Nothing to say

code:

```
class Solution {
public:
    bool isUgly(int num) {
        if(!num) return false;
```

```
        for(int m: {2, 3, 5}) while(num%m == 0) num /= m;
        return num == 1;
    }
};
```

803 (stuck)

- [ ] Re solve it

- BFS → TLE

302

goodspeed's solution:

lang: c++

idea: bfs

code:

```
class Solution {
    typedef pair<int, int> ii;
    int dx[4] = {1, -1, 0, 0};
    int dy[4] = {0, 0, 1, -1};
public:
    int minArea(vector<vector<char>>& image, int x, int y) {
        int l=x, r=x, u=y, d=y, n=image.size(), m=image[0].siz
e();
        queue<ii> Q;
        image[x][y] = '0';
        Q.push(ii(x, y));
        while(!Q.empty()){
            ii P = Q.front();
            Q.pop();
            l = min(l, P.first);
            r = max(r, P.first);
            d = min(d, P.second);
            u = max(u, P.second);
```

```
            for(int k=0;k<4;++k){
                x = P.first + dx[k];
                y = P.second + dy[k];
                if(x>=0 && x<n && y>=0 && y<m && image[x][y]=
='1'){
                    image[x][y] = '0';
                    Q.push(ii(x, y));
                }
            }
        }
        return (u-d+1) * (r-l+1);
    }
};
```

- [ ] Binary search solution @Zebo L
- [ ] DFS solution @Zebo L

460

goodspeed's solution:

lang: c++

idea: linked list

code:

```
#include<cassert>
struct Node{
    int key;
    int val;
    int freq;
    Node *next, *prev;
    Node(int k=0, int x=0, int f=0): key(k), val(x), freq(f),
next(NULL), prev(NULL) {}
};

Node *removeNode(Node *p){
```

```
    if(p){
        p->prev->next = p->next;
        if(p->next) p->next->prev = p->prev;
    }
    return p;
}

void insertNode(Node *pos, Node *p){
    if(!p) return;
    p->prev = pos;
    p->next = pos->next;
    if(pos->next) pos->next->prev = p;
    pos->next = p;
}


class LFUCache {
    unordered_map<int, Node*> pos;
    int C, cnt;
    Node lead, *last;
public:
    LFUCache(int capacity) {
        C = capacity;
        cnt = 0;
        last = &lead;
    }

    int get(int key) {
        if(!pos.count(key) || !C) return -1;
        ++pos[key]->freq;
```

```
        Node *p = pos[key]->prev;
        while(p->freq <= pos[key]->freq && p!=&lead) p=p->pre
v;
        if(p->next != pos[key]){
            pos[key] = removeNode(pos[key]);
            insertNode(p, pos[key]);
        }
        while(last->next) last = last->next;
        assert(!last->next);
        return pos[key]->val;
    }

    void put(int key, int value) {
        if(!C) return;
        Node *p;
        if(!pos.count(key)){
            if(cnt < C){
                pos[key] = new Node(key, value, 1);
                ++cnt;
            }
            else{
                pos.erase(last->key);
                pos[key] = removeNode(last);
                pos[key]->key = key;
                pos[key]->val = value;
                pos[key]->freq = 1;
                assert((int)pos.size() == C);
                last = pos[key]->prev;
            }
            p = last;
```

```
        }
        else{
            pos[key] = removeNode(pos[key]);
            pos[key]->val = value;
            ++pos[key]->freq;
            p = pos[key]->prev;
        }
        while(p->freq <= pos[key]->freq && p!=&lead) p=p->pre
v;
        if(p->next != pos[key]) insertNode(p, pos[key]);
        while(last->next) last = last->next;
        assert(!last->next);
        assert((int)pos.size() <= C);
    }
};
```

- [ ] Too slow, need revision @Zebo L

121

goodspeed's solution:

lang: c++

idea: one pass

code:

```
class Solution {
public:
    int maxProfit(vector<int>& prices) {
        int ans = 0, m = INT_MAX;
        for(int p: prices){
            m = min(p, m);
            ans = max(ans, p-m);
        }
        return ans;
```

```
    }
};
```

247

goodspeed's solution:

lang: c++

idea: brute force

code:

```
class Solution {
    const string all = "018";
    const string a = "01869";
    const string ra = "01896";
    const string b = "1869";
    const string rb = "1896";
public:
    vector<string> findStrobogrammatic(int n) {
        if(!n) return vector<string>();
        if(n==1) return vector<string>{"0", "1", "8"};
        vector<string> ans;
        int N = 4;
        for(int j=1; j<n/2; ++j) N*=5;
        if(n%2) N *= 3;
        for(int j=0; j<N; ++j){
            int k = j;
            string tmp(n, ' ');
            tmp[0] = b[k%4];
            tmp[n-1] = rb[k%4];
            k/=4;
            for(int j=1; j<n/2; ++j){
                tmp[j] = a[k%5];
                tmp[n-j-1] = ra[k%5];
```

```
                k/=5;
            }
            if(n%2) tmp[n/2] = all[k%3];
            ans.push_back(tmp);
        }
        sort(ans.begin(), ans.end());
        return ans;
    }
};
```