田船长算法刷题（二）

剑指 Offer 61 扑克牌中的顺子

```cpp
class Solution {
public:
    bool isStraight(vector<int>& nums) {
        sort(nums.begin(), nums.end());
        int cnt = 0;
        for (int i = 0; i < nums.size(); i++) {
            if (nums[i] == 0) cnt++;
        }
        for (int i = cnt + 1; i < nums.size(); i++) {
            if (nums[i] == nums[i - 1]) {
                return false;
            }
        }
        int mmax = nums[4];
        int mmin = cnt == 5 ? nums[4] : nums[cnt];
        if (mmax - mmin < 5) {
            return true;
        }
        return false;
    }
};
```

剑指 Offer II 008. 和大于等于 target 的最短子数组

```cpp
class Solution {
public:
    int minSubArrayLen(int target, vector<int>& nums) {
        int l = 0, now = 0, ans = INT_MAX;
        for (int i = 0; i < nums.size(); i++) {
            now += nums[i];
            while (now - nums[l] >= target) {
                now -= nums[l];
                l++;
            }
            if (now >= target) {
                ans = min(ans, i - l + 1);
            }
        }
        if (ans == INT_MAX) {
            return 0;
```

```
        }
        return ans;
    }
};
```

剑指 Offer II 009 乘积小于 K 的子数组

```
class Solution {
public:
    int numSubarrayProductLessThanK(vector<int>& nums, int k) {
        if (k <= 1) {
            return 0;
        }
        int l = 0, now = 1, ans = 0;
        for (int i = 0; i < nums.size(); i++) {
            now *= nums[i];
            while (now >= k) {
                now /= nums[l];
                l++;
            }
            ans += i - l + 1;
        }
        return ans;
    }
};
```

剑指 Offer II 014 字符串中的变位词

```
class Solution {
public:
    bool checkInclusion(string s1, string s2) {
        if (s1.size() > s2.size()) {
            return false;
        }
        int n1[130] = {0}, n2[130] = {0};
        for (int i = 0; i < s1.size(); i++) {
            n1[s1[i]]++;
            n2[s2[i]]++;
        }
        int cnt = 0;
        for (int i = 'a'; i <= 'z'; i++) {
            if (n1[i] != n2[i]) {
                cnt++;
            }
        }
```

```
        for (int i = s1.size(); i < s2.size(); i++) {
            if (cnt == 0) break;
            //新加入一个字符
            if (n1[s2[i]] == n2[s2[i]]) cnt++;
            n2[s2[i]]++;
            if (n1[s2[i]] == n2[s2[i]]) cnt--;
            //窗口前面移除一个字符
            if (n1[s2[i - s1.size()]] == n2[s2[i - s1.size()]]) cnt++;
            n2[s2[i - s1.size()]]--;
            if (n1[s2[i - s1.size()]] == n2[s2[i - s1.size()]]) cnt--;
        }
        if (cnt == 0) return true;
        return false;
    }
};
```

[剑指 Offer II 007. 数组中和为 0 的三个数]

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

剑指 Offer II 018 有效的回文

```cpp
class Solution {
public:
    int func(char &c) {
        if (c >= 'A' && c <= 'Z') {
            c += 'a' - 'A';
            return 0;
        }
        if (c >= 'a' && c <='z' || c >= '0' && c <= '9') {
            return 0;
        }
        return 1;
    }
    bool isPalindrome(string s) {
        int l = 0, r = s.size() - 1;
        while (l < r) {
            while (l < s.size() && func(s[l])) {
                l++;
            }
            while (r >= 0 && func(s[r])) {
                r--;
            }
            if (l == s.size() || r < 0) {
                return true;
            }
            if (s[l] != s[r]) {
                return false;
            }
            l++, r--;
        }
        return true;
    }
};
```

剑指 Offer II 019. 最多删除一个字符得到回文

```cpp
class Solution {
public:
    int func(string &s, int l, int r) {
        while (l < r) {
            if (s[l] != s[r]) {
                return false;
            }
            l++, r--;
```

```
        }
        return true;
    }
    bool validPalindrome(string s) {
        int l = 0, r = s.size() - 1;
        while (l < r) {
            if (s[l] == s[r]) {
                l++, r--;
            } else {
                return func(s, l, r - 1) || func(s, l + 1, r);
            }
        }
        return true;
    }
};
```

967 连续差相同的数字

```
class Solution {
public:
    //now当前选出的数字 left还剩几个数字要选
    void func(vector<int> &ans, int now, int left, int k) {
        if (left == 0) {
            ans.push_back(now);
            return ;
        }
        int t = now % 10 - k;
        if (t >= 0) {
            func(ans, now * 10 + t, left - 1, k);
        }
        t = now % 10 + k;
        if (k != 0 && t < 10) {
            func(ans, now * 10 + t, left - 1, k);
        }
    }
    vector<int> numsSameConsecDiff(int n, int k) {
        vector<int> ans;
        for (int i = 1; i < 10; i++) {
            func(ans, i, n - 1, k);
        }
        return ans;
    }
};
```

79 单词搜索

```cpp
class Solution {
public:
    int n, m;
    int dir[4][2] = {0, 1, 1, 0, 0, -1, -1, 0};
    int mark[10][10] = {0};
    int func(vector<vector<char> > &mmap, string &s, int now, int x, int y) {
        if (now == s.size()) {
            return true;
        }
        for (int i = 0; i < 4; i++) {
            int xx = x + dir[i][0];
            int yy = y + dir[i][1];
            if (xx < 0 || yy < 0 || xx == n || yy == m || mark[xx][yy] == 1)
continue;
            if (mmap[xx][yy] == s[now]) {
                mark[xx][yy] = 1;
                if (func(mmap, s, now + 1, xx, yy)) {
                    return true;
                }
                mark[xx][yy] = 0;
            }
        }
        return false;
    }
    bool exist(vector<vector<char>>& board, string word) {
        n =  board.size(), m = board[0].size();
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                if (board[i][j] == word[0]) {
                    mark[i][j] = 1;
                    if (func(board, word, 1, i, j)) {
                        return true;
                    }
                    mark[i][j] = 0;
                }
            }
        }
        return false;
    }
};
```