一、题目说明

题目76. Minimum Window Substring，求字符串S中最小连续字符串，包括字符串T中的所有字符，复杂度要求是O(n)。难度是Hard！

二、我的解答

先说我的思路：

（1）统计t中每个字符出现的次数，

（2）用hash存储s中出现t中字符的位置，

（3）计算最短字符串。

思路有了，惭愧的是，这个题目，我没做出来。

后来参考大神的实现了，用“滑动窗口”实现。

代码如下：

```
class Solution{
    public:
        string minWindow(string s,string t){
            if(s.empty() || t.empty()) return "";
            int left=0,right=0;
            string res = s;
            int minLen = INT_MAX;
            int start = 0;

            unordered_map<char,int> window;//当前「窗口」中包含的字符及出现的次数
            unordered_map<char,int> needs;//t 中包含的字符及出现次数


            //统计字符串t中各个字符的数量
            for(char ch:t){
                needs[ch]++;//如果该 key不存在，C++ 会自动创建这个 key，并把 map[key]
赋值为 0
            }

            int match = 0;//记录 window 中已经有多少字符符合要求了

            while(right<s.size()){
                char c1 = s[right];
                if(needs.count(c1)){
                    window[c1]++;
                    if(window[c1]==needs[c1]){
                        match++;//字符 c1 的出现次数符合要求了
                    };
                }
                right++;

                //window 中的字符串已符合 needs 的要求了
                while(match==needs.size()){
                    //缩减res
                    if(right-left<minLen){
```

```
                        start = left;
                        minLen = right - left;
                    }
                    char c2 = s[left];
                    if(needs.count(c2)){
                        window[c2]--;
                        if(window[c2]<needs[c2]){
                            match--;
                        }
                    }
                    left++;
                }
            }

            return minLen == INT_MAX ? "" : s.substr(start, minLen);

        }

};
```

性能一般：

```
Runtime: 28 ms, faster than 45.63% of C++ online submissions for Minimum Window Substring.
Memory Usage: 10.2 MB, less than 68.00% of C++ online submissions for Minimum Window Substring.
```

三、优化措施

我消化消化再说。