**[Minimum Window Substring](https://leetcode.com/problems/minimum-window-substring/)**

Given two strings s and t of lengths m and n respectively, return *the****minimum window***

***substring***

*of*s*such that every character in*t*(****including duplicates****) is included in the window*. If there is no such substring, return *the empty string*"".

The testcases will be generated such that the answer is **unique**.

**Example 1:**

**Input:** s = "ADOBECODEBANC", t = "ABC"

**Output:** "BANC"

**Explanation:** The minimum window substring "BANC" includes 'A', 'B', and 'C' from string t.

**Example 2:**

**Input:** s = "a", t = "a"

**Output:** "a"

**Explanation:** The entire string s is the minimum window.

**Example 3:**

**Input:** s = "a", t = "aa"

**Output:** ""

**Explanation:** Both 'a's from t must be included in the window.

Since the largest window of s only has one 'a', return empty string.

**Constraints:**

* m == s.length
* n == t.length
* 1 <= m, n <= 105
* s and t consist of uppercase and lowercase English letters.

**Code**

class Solution {

public:

    string minWindow(string s, string t) {

        if (s.empty() || t.empty()) {

            return "";

        }

        unordered\_map<char, int> dictT;

        for (char c : t) {

            int count = dictT[c];

            dictT[c] = count + 1;

        }

        int required = dictT.size();

        int l = 0, r = 0;

        int formed = 0;

        unordered\_map<char, int> windowCounts;

        int ans[3] = { -1, 0, 0 };

        while (r < s.length()) {

            char c = s[r];

            int count = windowCounts[c];

            windowCounts[c] = count + 1;

            if (dictT.find(c) != dictT.end() && windowCounts[c] == dictT[c]) {

                formed++;

            }

            while (l <= r && formed == required) {

                c = s[l];

                if (ans[0] == -1 || r - l + 1 < ans[0]) {

                    ans[0] = r - l + 1;

                    ans[1] = l;

                    ans[2] = r;

                }

                windowCounts[c]--;

                if (dictT.find(c) != dictT.end() && windowCounts[c] < dictT[c]) {

                    formed--;

                }

                l++;

            }

            r++;

        }

        return ans[0] == -1 ? "" : s.substr(ans[1], ans[0]);

    }

};

Link : <https://leetcode.com/problems/minimum-window-substring/?envType=daily-question&envId=2024-02-04>