76. Minimum Window Substring

Hard 9547 517 Add to List Share

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**.

A **substring** is a contiguous sequence of characters within the string.

smallest substring of 's' which contains all the chars of 't'.

```
Input: s = "ADOBECODEBANC", t = "ABC"
Output: "BANC"
Explanation: The minimum window substring "BANC"
includes 'A', 'B', and 'C' from string t.
```

$O(n^3)$ (repeat i to 'j')

↓

$O(n^2)$ (aquire)

↓

$O(n)$ (aquire & release)

s = a b d e c b a k b a

t = a a b b

mtc = ~~ϕ~~ ~~x~~ ~~xy~~ ~~xyz~~ ~~xyzk~~ ~~xyzkb~~ xyzkba

j

a b d e c b a k b a

0 1 2 3 4 5 6 7 8 9

i

aabb

a-2

b-2

map a-2

b-1 k-1

i → aquire

j → release

ans = bakba

while ( ) {

(i) aquire : mtc < t.length

(ii) release : mtc == t.length

}

mtc = ~~0 1 2 3 4 5~~ 4 3

j

a b d e c b a k b a

0 1 2 3 4 5 6 7 8 9

i

t : aabb

a - 2

b - 2

map : a - 2

b - 1 k - 1

bakba

~~cbakba~~

ans : ~~abdecba~~

while (i < s.length()-1){

aquire

```
while(i < s.length()-1 && mtc < t.length()) {
    i++;

    //aquire ith char
    char ch = s.charAt(i);

    int nf = sm.getOrDefault(ch,0) + 1;
    sm.put(ch,nf);

    //impact on mtc
    if(sm.get(ch) <= tm.getOrDefault(ch,0)) {
        mtc++;
    }
}
```

release

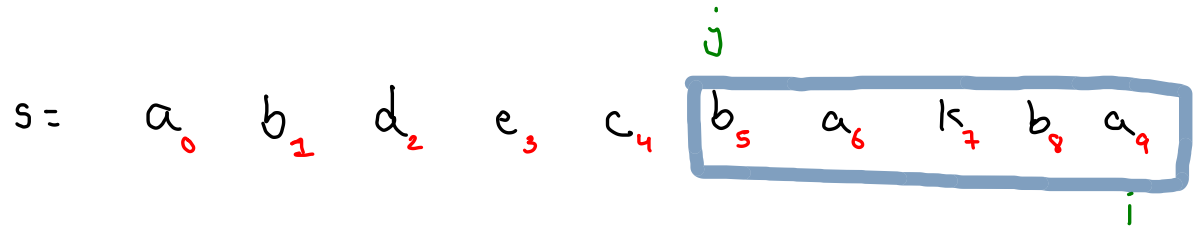
```
//release
while(j < i && mtc == t.length()) {
    //ans -> j+1 to i

    int len = i-j;
    if(len < olen) {
        as = j+1;
        ae = i;
        olen = len;
    }

    j++;

    //release jth char
    char ch = s.charAt(j);
    if(sm.get(ch) == 1) {
        sm.remove(ch);
    }
    else {
        int nf = sm.get(ch) - 1;
        sm.put(ch,nf);
    }

    //impact on match count
    if(sm.getOrDefault(ch,0) < tm.getOrDefault(ch,0)) {
        mtc--;
    }


}
```

}

mtc = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ 3

as = 5

ae = 9



sm a-2

b-1 k-1

t = aabb

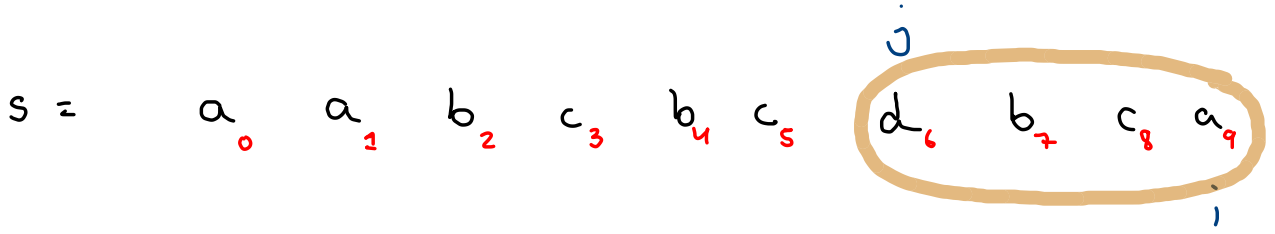
tm: a-2

b-2

Smallest Substring Of A String Containing All Unique Characters Of Itself

as = ~~-1~~ ~~0~~ ~~1~~ ~~5~~ 6

ae = ~~-1~~ ~~5~~ 9



1. aquire → map.size() < hs.size()

2. release → map.size() == hs.size()

Smallest Substring Of A String Containing All Unique Characters Of Itself

```
while(i < s.length()-1) {
    while(i < s.length()-1 && map.size() < hs.size()) {
        i++;

        char ch = s.charAt(i);
        int nf = map.getOrDefault(ch,0) + 1;

        map.put(ch,nf);
    }

    while(j < i && map.size() == hs.size()) {
        //ans
        int len = i-j;

        if(len < olen) {
            olen = len;
        }

        j++;

        char ch = s.charAt(j);

        if(map.get(ch) == 1) {
            map.remove(ch);
        }
        else {
            int nf = map.get(ch) - 1;
            map.put(ch,nf);
        }
    }
}
```

olen = ∞ ≠ 6 s

s : a0 b1 b2 c3 a4 c5 d6 a7

j i

map a-2 c-2 d-1

a, b, c, d