Given strings s1 and s2, return *the minimum contiguous substring part of* s1*, so that* s2 *is a subsequence of the part*.

If there is no such window in s1 that covers all characters in s2, return the empty string "". If there are multiple such minimum-length windows, return the one with the **left-most starting index**.

**Example 1:**

Input: s1 = "abcdebdde", s2 = "bde"  
Output: "bcde"  
Explanation:   
"bcde" is the answer because it occurs before "bdde" which has the same length.  
"deb" is not a smaller window because the elements of s2 in the window must occur in order.

**Example 2:**

Input: s1 = "jmeqksfrsdcmsiwvaovztaqenprpvnbstl", s2 = "u"  
Output: ""

**Constraints:**

* 1 <= s1.length <= 2 \* 104
* 1 <= s2.length <= 100
* s1 and s2 consist of lowercase English letters.