Minimum Window Subsequence

Try to solve the Minimum Window Subsequence problem.

We'll cover the following

- Statement
- Examples
- · Understand the problem
- Figure it out!
- Try it yourself

Statement

Given two strings, str1 and str2, find the shortest substring in str1 such that str2 is a subsequence of that substring.

A **substring** is defined as a contiguous sequence of characters within a string. A **subsequence** is a sequence that can be derived from another sequence by deleting zero or more elements without changing the order of the remaining elements.

Let's say you have the following two strings:

```
str1 = "abbcb"
```

str2 = "ac"

In this example, "abbc" is a substring of str1, from which we can derive str2 simply by deleting both the instances of the character b. Therefore, str2 is a subsequence of this substring. Since this substring is the shortest among all the substrings in which str2 is present as a subsequence, the function should return this substring, that is, "abbc".

If there is no substring in str1 that covers all characters in str2, return an empty string.

If there are multiple minimum-length substrings that meet the subsequence requirement, return the one with the left-most starting index.

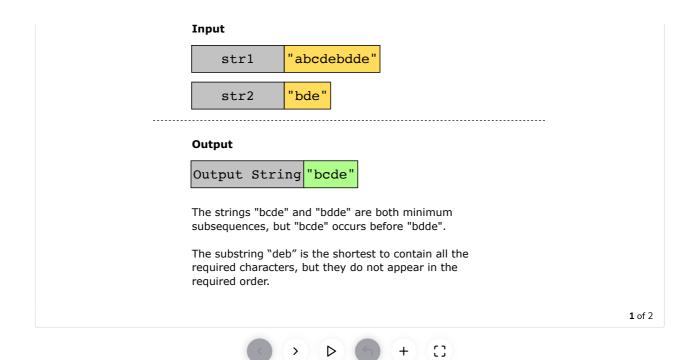
Constraints:

- $1 \le \text{str1.length} \le 2 \times 10^3$
- $1 \le \text{str2.length} \le 100$
- str1 and str2 consist of uppercase and lowercase English letters.

Examples

Тт

?



Understand the problem

Let's take a moment to make sure you've correctly understood the problem. The quiz below helps you check if you're solving the correct problem:

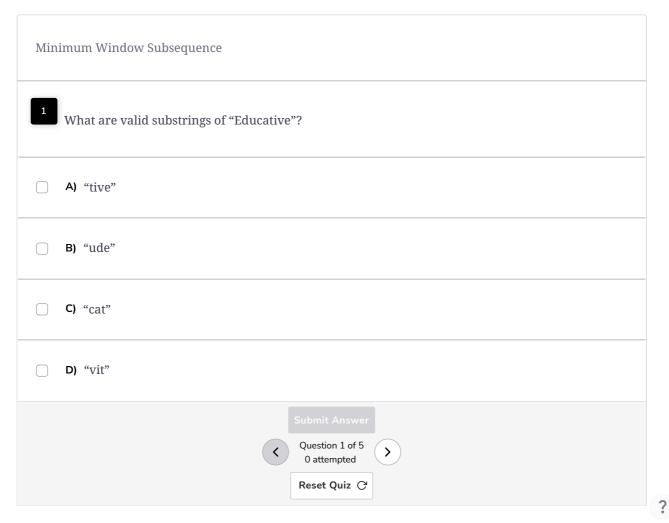


Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

Tτ

6



Try it yourself

Implement your solution in main.java in the following coding playground:

