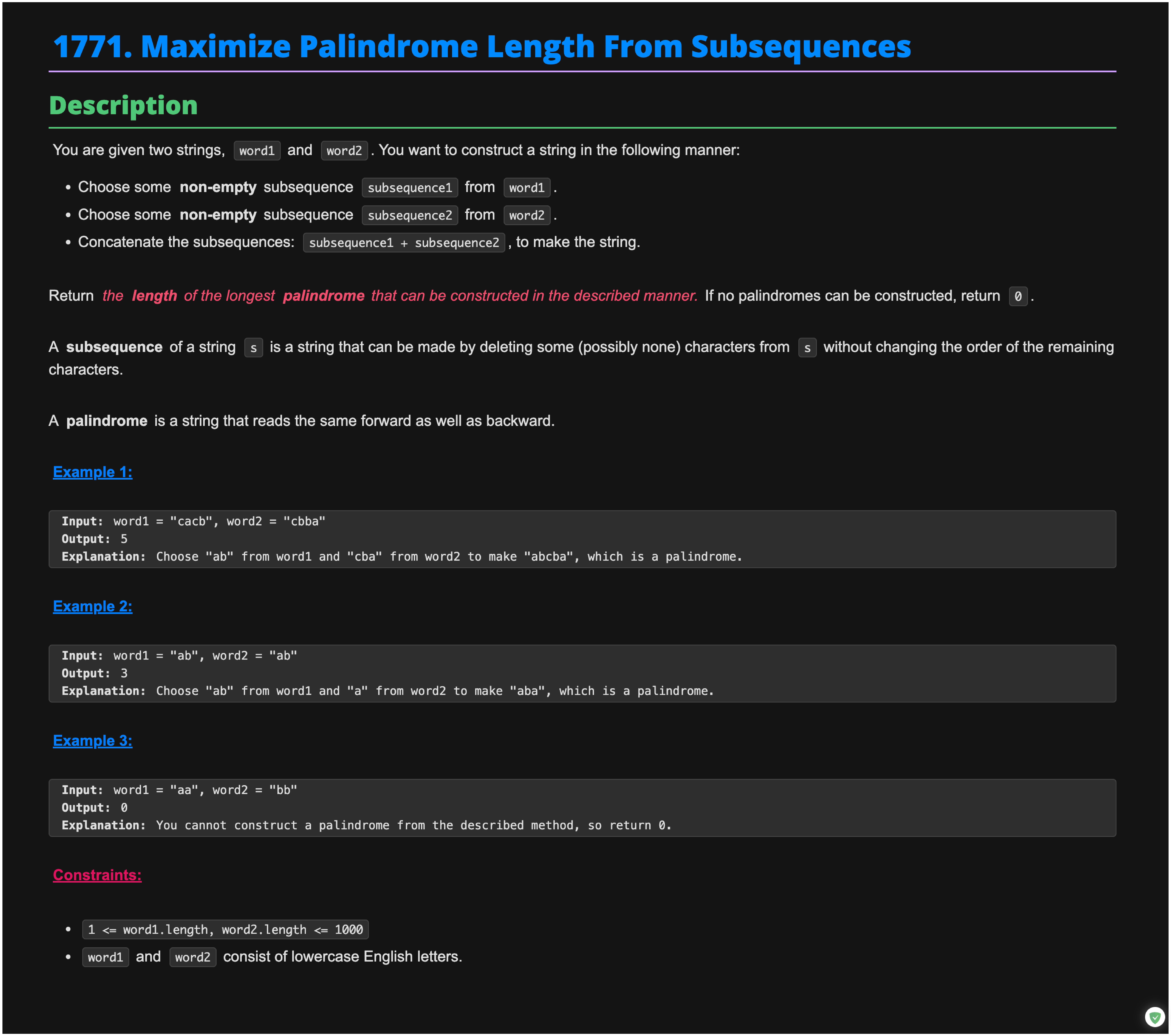

1771. Maximize Palindrome Length From Subsequences

Description

You are given two strings, `word1` and `word2`. You want to construct a string in the following manner:

- Choose some **non-empty** subsequence `subsequence1` from `word1`.
- Choose some **non-empty** subsequence `subsequence2` from `word2`.
- Concatenate the subsequences: `subsequence1 + subsequence2`, to make the string.

Return *the **length** of the longest **palindrome** that can be constructed in the described manner.* If no palindromes can be constructed, return `0`.

A **subsequence** of a string `s` is a string that can be made by deleting some (possibly none) characters from `s` without changing the order of the remaining characters.

A **palindrome** is a string that reads the same forward as well as backward.

Example 1:

```
Input: word1 = "cacb", word2 = "cbba"
Output: 5
Explanation: Choose "ab" from word1 and "cba" from word2 to make "abcba", which is a palindrome.
```

Example 2:

```
Input: word1 = "ab", word2 = "ab"
Output: 3
Explanation: Choose "ab" from word1 and "a" from word2 to make "aba", which is a palindrome.
```

Example 3:

```
Input: word1 = "aa", word2 = "bb"
Output: 0
Explanation: You cannot construct a palindrome from the described method, so return 0.
```

Constraints:

- `1 <= word1.length, word2.length <= 1000`
- `word1` and `word2` consist of lowercase English letters.