Ex1: You are given two strings word1 and word2. Merge the strings by adding letters in alternating order, starting with word1. If a string is longer than the other, append the additional letters onto the end of the merged string.

Return *the merged string.*

**Example 1:**

**Input:** word1 = "abc", word2 = "pqr"

**Output:** "apbqcr"

**Explanation:** The merged string will be merged as so:

word1: a b c

word2: p q r

merged: a p b q c r

**Example 2:**

**Input:** word1 = "ab", word2 = "pqrs"

**Output:** "apbqrs"

**Explanation:** Notice that as word2 is longer, "rs" is appended to the end.

word1: a b

word2: p q r s

merged: a p b q r s

**Example 3:**

**Input:** word1 = "abcd", word2 = "pq"

**Output:** "apbqcd"

**Explanation:** Notice that as word1 is longer, "cd" is appended to the end.

word1: a b c d

word2: p q

merged: a p b q c d

**Constraints:**

* 1 <= word1.length, word2.length <= 100
* word1 and word2 consist of lowercase English letters.

Ex2: For two strings s and t, we say "t divides s" if and only if s = t + t + t + ... + t + t (i.e., t is concatenated with itself one or more times).

Given two strings str1 and str2, return *the largest string*x*such that*x*divides both*str1*and*str2.

**Example 1:**

**Input:** str1 = "ABCABC", str2 = "ABC"

**Output:** "ABC"

**Example 2:**

**Input:** str1 = "ABABAB", str2 = "ABAB"

**Output:** "AB"

**Example 3:**

**Input:** str1 = "LEET", str2 = "CODE"

**Output:** ""

**Constraints:**

* 1 <= str1.length, str2.length <= 1000
* str1 and str2 consist of English uppercase letters.