# 2546. Apply Bitwise Operations to Make Strings Equal

## Description

You are given two **0-indexed binary** strings `s` and `target` of the same length `n`. You can do the following operation on `s` **any** number of times:

- Choose two **different** indices `i` and `j` where `0 <= i, j < n`.
- Simultaneously, replace `s[i]` with ( `s[i]` **OR** `s[j]` ) and `s[j]` with ( `s[i]` **XOR** `s[j]` ).

For example, if `s = "0110"`, you can choose `i = 0` and `j = 2`, then simultaneously replace `s[0]` with ( `s[0]` **OR** `s[2]` = `0` **OR** `1` = `1` ), and `s[2]` with ( `s[0]` **XOR** `s[2]` = `0` **XOR** `1` = `1` ), so we will have `s = "1110"`.

Return `true` *if you can make the string* `s` *equal to* `target`, *or* `false` *otherwise*.

### Example 1:

```
Input: s = "1010", target = "0110"
Output: true
Explanation: We can do the following operations:
- Choose i = 2 and j = 0. We have now s = " 0 0 1 0".
- Choose i = 2 and j = 1. We have now s = "0 11 0".
Since we can make s equal to target, we return true.
```

### Example 2:

```
Input: s = "11", target = "00"
Output: false
Explanation: It is not possible to make s equal to target with any number of operations.
```

### Constraints:

- `n == s.length == target.length`
- `2 <= n <= 10^5`
- `s` and `target` consist of only the digits `0` and `1`.