

927. Three Equal Parts

Description

You are given an array `arr` which consists of only zeros and ones, divide the array into **three non-empty parts** such that all of these parts represent the same binary value.

If it is possible, return any `[i, j]` with `i + 1 < j`, such that:

- `arr[0], arr[1], ..., arr[i]` is the first part,
- `arr[i + 1], arr[i + 2], ..., arr[j - 1]` is the second part, and
- `arr[j], arr[j + 1], ..., arr[arr.length - 1]` is the third part.
- All three parts have equal binary values.

If it is not possible, return `[-1, -1]`.

Note that the entire part is used when considering what binary value it represents. For example, `[1,1,0]` represents `6` in decimal, not `3`. Also, leading zeros **are allowed**, so `[0,1,1]` and `[1,1]` represent the same value.

Example 1:

Input: `arr = [1,0,1,0,1]`
Output: `[0,3]`

Example 2:

Input: `arr = [1,1,0,1,1]`
Output: `[-1,-1]`

Example 3:

Input: `arr = [1,1,0,0,1]`
Output: `[0,2]`

Constraints:

- `3 <= arr.length <= 3 * 104`
- `arr[i]` is `0` or `1`

