1562. Find Latest Group of Size M

Description

Given an array `arr` that represents a permutation of numbers from `1` to `n`.

You have a binary string of size `n` that initially has all its bits set to zero. At each step `i` (assuming both the binary string and `arr` are 1-indexed) from `1` to `n`, the bit at position `arr[i]` is set to `1`.

You are also given an integer `m`. Find the latest step at which there exists a group of ones of length `m`. A group of ones is a contiguous substring of `1`'s such that it cannot be extended in either direction.

Return *the latest step at which there exists a group of ones of length* ***exactly*** `m`. *If no such group exists, return* `-1`.

Example 1:

```
Input: arr = [3,5,1,2,4], m = 1
Output: 4
Explanation:
Step 1: "00 1 00", groups: ["1"]
Step 2: "0010 1", groups: ["1", "1"]
Step 3: " 1 0101", groups: ["1", "1", "1"]
Step 4: "1 1 101", groups: ["111", "1"]
Step 5: "111 1 1", groups: ["11111"]
The latest step at which there exists a group of size 1 is step 4.
```

Example 2:

```
Input: arr = [3,1,5,4,2], m = 2
Output: -1
Explanation:
Step 1: "00 1 00", groups: ["1"]
Step 2: " 1 0100", groups: ["1", "1"]
Step 3: "1010 1", groups: ["1", "1", "1"]
Step 4: "101 1 1", groups: ["1", "111"]
Step 5: "1 1 111", groups: ["11111"]
No group of size 2 exists during any step.
```

Constraints:

- `n == arr.length`
- `1 <= m <= n <= 10^5`
- `1 <= arr[i] <= n`
- All integers in `arr` are **distinct**.