1310. XOR Queries of a Subarray

You are given an array `arr` of positive integers. You are also given the array `queries` where `queries[i] = [left`$_i$`, right`$_i$`]`.

For each query$_i$ compute the **XOR** of elements from `left`$_i$ to `right`$_i$ (that is, `arr[left`$_i$`] XOR arr[left`$_i$` + 1] XOR ... XOR arr[right`$_i$`]`).

Return an array `answer` where `answer[i]` is the answer to the `i`th query.

Example 1:

```
Input: arr = [1,3,4,8], queries = [[0,1],[1,2],[0,3],[3,3]]
Output: [2,7,14,8]
Explanation:
The binary representation of the elements in the array are:
1 = 0001
3 = 0011
4 = 0100
8 = 1000
The XOR values for queries are:
[0,1] = 1 xor 3 = 2
[1,2] = 3 xor 4 = 7
[0,3] = 1 xor 3 xor 4 xor 8 = 14
[3,3] = 8
```

Example 2:

```
Input: arr = [4,8,2,10], queries = [[2,3],[1,3],[0,0],[0,3]]
Output: [8,0,4,4]
```

Constraints:

- `1 <= arr.length, queries.length <= 3 * 10`^{4}
- `1 <= arr[i] <= 10`^{9}
- `queries[i].length == 2`
- `0 <= left`$_i$ `<= right`$_i$ `< arr.length`

1310. XOR Queries of a Subarray

```
/*
    Prefix xor array
    Time complexity: O(n+n+Q)=O(n+Q)
    Space complexity: O(n)
    n: size of the array
    Q: number of queries
*/
typedef std::vector<int> vi;
typedef std::vector<vi> vvi;
class Solution {
public:
    vi xorQueries(vi& arr, vvi& queries){
        int n=arr.size();

        vi pre(n+1,0);
        for(int i=0;i<n;++i) pre[i+1]=pre[i]^arr[i];

        vi ans;
        for(auto& q: queries){
            int left=q[0];
            int right=q[1];
            ans.push_back(pre[right+1]^pre[left]);
        }

        return ans;
    }
};
```