Dominating XOR

For an array *arr* of *n* positive integers, count the unordered pairs (i, j) (0 ≤ i < j < n) where *arr[i] XOR arr[j] > arr[i]AND arr[j].* XOR denotes the bitwise XOR operation and AND denotes the bitwise AND operation.

**Example**

Given *n* = 4, *arr* = [4, 3, 5, 2]. All unordered pairs (i, j) are-

Indices XOR AND XOR > AND   
(0,1)    7    0    True   
(0,2)    1    4    False   
(0,3)    6    0    True   
(1,2)    6    1    True   
(1,3)    1    2    False   
(2,3)    7    0    True 

For the first line:

* *arr[0] = 4, arr[1] = 3*
* *4 XOR 3 = 7*
* *4 AND 3 = 0*
* *7 > 3*

There are 4 good pairs where XOR > AND shows True. Return 4.

**Function Description**

Complete the function *dominatingXorPairs* in the editor below.

*dominatingXorPairs* has the following parameter:

*int arr[n]:*  an array of integers

**Returns**

*long**int:* the number of good pairs

**Constraints**

* 1 ≤ *n* ≤ 105
* 1 ≤ *arr[i]* < 230

**Sample Case 0**

**Sample Input for Custom Testing**

STDIN    FUNCTION   
-----    --------   
4     →   n = 4   
1     →   arr = [1, 1, 5, 7]    
1   
5   
7 

**Sample Output**

4

**Explanation**

Indices XOR AND XOR > AND   
(0,1)    0    1    False   
(0,2)    4    1    True   
(0,3)    6    1    True   
(1,2)    4    1    True   
(1,3)    6    1    True   
(2,3)    2    5    False

**Sample Case 1**

**Sample Input for Custom Testing**

STDIN    FUNCTION   
-----    --------   
3     → n = 3   
1     →  arr = [1, 7, 2]   
7   
2

**Sample Output**

3

**Explanation**

All unordered pairs satisfy the condition.