1. Bitwise AND

Given an array of non-negative integers, count the number of unordered pairs of array elements such that their bitwise AND is a power of 2.

For example, let's say the array is *arr = [10, 7, 2, 8, 3]*, and let '&' denote the bitwise AND operator. There are 6 unordered pairs of its elements that have a bitwise AND that is a power of two:

- For indices (0,1), 10 & 7 = 2, which is a power of 2.
- For indices (0,2), 10 & 2 = 2, which is a power of 2.
- For indices (0,3), 10 & 8 = 8, which is a power of 2.
- For indices (0,4), 10 & 3 = 2, which is a power of 2.
- For indices (1,2), 7 & 2 = 2, which is a power of 2.
- For indices (2,4), 2 & 3 = 2, which is a power of 2.

Therefore, the answer is 6.

Function Description
Complete the function *countPairs* in the editor below.

countPairs has the following parameter:
  int *arr[n]:* an array of integers
Returns:
  int: the number of unordered pairs of elements of *arr* such that their bitwise AND is a power of 2

Constraints

- $1 \le n \le 2*10^5$
- $0 \le arr[i] < 2^{12}$

▼ Input Format For Custom Testing

The first line contains an integer, *n*, denoting the number of elements in *arr*.
Each line *i* of the *n* subsequent lines (where $0 \le i < n$) contains an integer describing *arr[i]*.

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN       Function
-----       --------
4     =>    n = 4
1     =>    arr = [1, 2, 1, 3]
2
1
3
```

Sample Output

```
4
```

Explanation
All unordered pair of elements whose bitwise AND is a power of 2 are:

- For indices (0,2), 1 & 1 = 1, which is a power of 2.
- For indices (0,3), 1 & 3 = 1, which is a power of 2.
- For indices (1,3), 2 & 3 = 2, which is a power of 2.
- For indices (2,3), 1 & 3 = 1, which is a power of 2.

Therefore, the answer is 4.

▼ Sample Case 1

Sample Input For Custom Testing

```
3
0
2
4
```

Sample Output

```
0
```

Explanation

There are no pairs of array elements such that their bitwise AND is a power of 2. Therefore, the answer is 0.