https://course.acciojob.com/idle?question=86aaf8d4-7a31-4f05-b4cc-37d8d545460b

- HARD
- Max Score: 50 Points

Triples with Bitwise AND Equal To Zero

Given an integer array `nums`, return the number of `AND` triples.

An `AND` triple is a triple of indices `(i, j, k)` such that:

0 <= i < nums.length

0 <= j < nums.length

0 <= k < nums.length

`nums[i] & nums[j] & nums[k] == 0`, where `&` represents the bitwise-AND operator.

Input Format

The first line of input contains a single integer `n`.

The next `n` line of input contains `n` space separated integers.

Output Format

Your task is to return the number of `AND` triples.

Example 1

Input

```
3
2 1 3
```

Output

```
12
```

Explanation

```
(i=0, j=0, k=1) : 2 & 2 & 1
(i=0, j=1, k=0) : 2 & 1 & 2
(i=0, j=1, k=1) : 2 & 1 & 1
(i=0, j=1, k=2) : 2 & 1 & 3
(i=0, j=2, k=1) : 2 & 3 & 1
(i=1, j=0, k=0) : 1 & 2 & 2
(i=1, j=0, k=1) : 1 & 2 & 1
(i=1, j=0, k=2) : 1 & 2 & 3
(i=1, j=1, k=0) : 1 & 1 & 2
(i=1, j=2, k=0) : 1 & 3 & 2
(i=2, j=0, k=1) : 3 & 2 & 1
(i=2, j=1, k=0) : 3 & 1 & 2
```

Example 2

Input

```
3
0 0 0
```

Output

```
27
```

Explanation

```
As all the number are `0`, we have 3^3,toatl number of solutions that is 27.
```

Constraints

1 <= n <= 1000

1 <= nums[i] <= 2^{16}

Topic Tags

- Hashing
- Bit Manipulation

My code

```
import java.util.*;

public class Main{
    public static void main(String[] args)
    {
        int n;
        Scanner in = new Scanner(System.in);
                n = in.nextInt();
        int nums[]=new int[n];
        for(int i=0;i<n;i++)
        nums[i]=in.nextInt();
        Solution obj=new Solution();
        System.out.println(obj.solve(nums));

    }

}

class Solution{

    static int solve(int nums[])
    {
        // your code here
                        // int n=nums.length();
                        // HashMap<integer,Integer>hm=new HashMap<>();

                        // for(int i=0;i<n;i++)
                        //          for(int j=0;j<n;j++)
                        //                  {
                        //
hm.put((nums[i]&nums[j]),hm.getOrDefault((nums[i]&nums[j]),0)+1);
                        //                  }
  //        for(int i=0;i<n;i++)
                        //                  {
                        //                              for(int j)
                        //                                      {

                        //                                      }
                        //                  }
                        int[] count = new int[1 << 16];
    for(int a: nums) for(int b: nums) count[a & b]++;
    int res = 0;
    for(int a: nums) for(int i = 0; i < count.length; i++) {
            if((a & i) == 0) res += count[i];
            else i += (a & i) - 1;
    }
    return res;
}
}
```