Problem: We define $f(x,y)$ as no. of different corresponding bits in the binary representation of x and y. for example.

$f(2,7)=2$, since binary representation of 2 and 7 are 010 and 111 respectively. The first and the third bit differ so $f(2,7)=2$.

You are given an array of N positive integers. $A1, A2 \ldots$ Ar. find sum of $f(Ai, Aj)$ for all pairs (i,j) such that $1 \le i, j \le N$. Return the answer modulo 10^9+7.

Example Input

Input 1:
$A=[1,3,5]$

Input 2:
$A=[2,3]$

Example Output

Output 1: 8

Output 2: 2

Algorithm

1. Initialize total = 0
2. Loop over all pairs
3. Compute XOR and count bits
4. Add count to total
5. Return total % (10^9+7)

Code

```
#include <bits/stdc++.h>
using namespace std;
int sumbitdifferences(vector<int>&arr) {
    const int MOD = 1e9+7;
    long long n = arr.size();
    long long total = 0;
for(int i=0; i<n; i++) {
    for(int j=i+1; j<n; j++) {
    int x = arr[i] ^ arr[j];
```

```
int count = 0;
while (n > 0) {
    if (n % 2 == 1) {
        count++;
    }
    n = n / 2;
}
```

```
if (n & 1) {
    count++;
}
n = n >> 1;
```

```
}
total = (total + count) % MOD;
}
return total;
}
int main() {
    vector<int> arr = {1, 3, 5};
    cout << sumBitDifference(arr);
}
```