Sets and Integers

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 256 megabytes

For any two non-negative integers x and y.

We say that x is a *supermask* of y if and only if $(x \,|\, y) = x$ (where $|$ denotes the bitwise OR operation - see notes for more clarification).

You are given an array of n non-negative integers $a_1, a_2, \ldots, a_n$. In one operation, you can choose two elements and replace them with any number that is a supermask of both elements. More formally, choose any two indices i, j such that $1 \leq i < j \leq n$, and replace a_i and a_j with an integer x where x is a supermask of both a_i and a_j.

You keep doing this operation until the array contains one element (that is, do it $n - 1$ times). Find the minimum possible value of such element.

Input

The first line of input starts with an integer t ($1 \leq t \leq 10^4$), the number of test cases.

The first line of each test case contains an integer n ($2 \leq n \leq 10^5$), the size of the array.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i < 2^{30}$).

The sum of n over all test cases doesn't exceed $5 \cdot 10^5$.

Output

For each test case, output one line containing one integer, the minimum possible value achievable by doing the described operation $n - 1$ times.

Example

standard input	standard output
2 4 5 4 2 6 5 1 2 4 8 16	7 31

Note

In the first test case, you can replace 5 and 6 with 7, since $5 = 101$ and $6 = 110$ and $7 = 111$ which is a supermask of both since $(5|7) = 7$ and $(6|7) = 7$. So, the array becomes $7, 4, 2$. Afterwards, you can replace 7 and 4 with 7, and 7 and 2 with 7. It can be proven that this is the minimum possible value achievable.

In the second test case, you can replace 1 and 16 with 17, and 2 and 4 with 6 making the array $6, 8, 17$. Afterwards you can replace 6 and 17 with 23 and 23 and 8 with 31. It can be proven that this is the minimum possible value achievable.

The OR of two bits x and y is 0 if and only if both x and y are 0. So, for example, 1 OR 0 is 1.

The bitwise OR of any two integers x and y can be found by converting each integer into binary and taking the OR of each two corresponding bits. For example, $(5|3) = 7$ since 3 in binary is `011` and 5 in binary is `101`, so the result is `111` which is 7.