Gemcrate

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Noir has n gems. The i-th gem has a positive integer a_i written on it. Noir wants to divide these gems into several non-empty groups. Each gem belongs to exactly one group.

Suppose the i-th group contains gems with label $k_{i,1}, k_{i,2}, \ldots, k_{i,p}$, then Noir treats the *brightness* of the i-th group as $a_{k_{i,1}} \oplus a_{k_{i,2}} \oplus \cdots \oplus a_{k_{i,p}}$, where $\oplus$ is the bitwise-XOR operation. Denote the *brightness* of the i-th group as B_i.

For a grouping method of m groups, Noir treats the *value* of this method as $B_1 \& B_2 \& \ldots \& B_m$, where & is the bitwise-AND operation.

Noir wants to find the maximum possible *value* over all grouping methods.

Input

The input contains multiple testcases. The first line of the input contains an integer T ($1 \le T \le 10^4$), the number of testcases.

For each testcase, the first line contains an integer n ($1 \le n \le 5 \times 10^5$), the number of gems.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($1 \le a_i < 2^{60}$), the integers written on gems.

It's guaranteed that the sum of n over all testcases does not exceed 5×10^5.

Output

For each testcase, print an integer representing the maximum possible *value* over all grouping methods.

Example

standard input	standard output
4 4 1 2 3 1 6 4 7 5 2 6 3 4 14 15 9 18 2 251508091405 13011908091815	2 6 26 13121614001578

Note

For the first testcase, a possible grouping method is $[1,2,3,1] = [1,3],[2,1]$ with a value of $B_1 \& B_2 = (1 \oplus 3) \& (2 \oplus 1) = 2 \& 3 = 2$. Another possible grouping method is $[1,2,3,1] = [1,2,3,1]$, with a lower value $B_1 = 1 \oplus 2 \oplus 3 \oplus 1 = 1$. It can be proved that it's not possible to achieve a value greater than 2.

For the second testcase, the best grouping method is $[4,7,5,2,6,3] = [7],[5,3],[6],[4,2]$ with a value of $7 \& (5 \oplus 3) \& 6 \& (4 \oplus 2) = 6$.