Lowbit OR Lowbit

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

小 W 有一个大小为 n 的正整数多重集合 $S = \{a_1, a_2, \ldots, a_n\}$。

接下来，他需要不断进行如下操作，直到集合中只剩下一个数：

- 每次选择集合中的两个元素，设它们的值分别为 x, y，将它们从集合中删除，并向集合中加入一个新数 $x \operatorname{or} y$。本次操作的代价为 $\operatorname{lowbit}(x) \operatorname{or} \operatorname{lowbit}(y)$。
- 其中 or 表示按位或运算；对于正整数 x，$\operatorname{lowbit}(x)$ 表示能够整除 x 的最大的二的幂。等价地，设 k 是满足 x 被 2^k 整除且 x 不被 2^{k+1} 整除的非负整数，则 $\operatorname{lowbit}(x) = 2^k$。例如：$\operatorname{lowbit}(12) = 4$, $\operatorname{lowbit}(10) = 2$。

请你帮小 W 求出所有方案中使集合最终合并成一个数的最小总代价。

Input

第一行包含一个整数 T $(1 \le T \le 10^4)$ —表示数据组数，每组数据输入格式如下：

- 对于每组数据，第一行包含一个整数 n $(1 \le n \le 2 \cdot 10^5)$，表示给定初始集合的大小。
- 第二行包含 n 个正整数 $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^9)$，表示给定的初始集合中的每个数。

保证所有数据中 $\sum n \le 4 \cdot 10^5$ 。

Output

对于每组测试数据，输出一行一个整数，表示最小总代价。

Example

standard input	standard output
4 3 1 2 4 4 6 10 14 22 5 3 5 6 8 12 6 2 3 4 7 16 24	8 6 18 35

Note

对于第一组数据，初始集合为 $\{1, 2, 4\}$，接下来：

- 可以先合并 1 和 2，代价为 $\operatorname{lowbit}(1) \operatorname{or} \operatorname{lowbit}(2) = 3$；合并后得到 $1 \operatorname{or} 2 = 3$，集合变为 $\{3, 4\}$。
- 再合并 3 和 4，代价为 $\operatorname{lowbit}(3) \operatorname{or} \operatorname{lowbit}(4) = 5$；合并后只剩下一个数。

总代价为 $3 + 5 = 8$，可以证明不存在更小的总代价。