You are given a **0-indexed** integer array nums. In one operation, select **any** non-negative integer x and an index i, then **update** nums[i] to be equal to nums[i] AND (nums[i] XOR x).

Note that AND is the bitwise AND operation and XOR is the bitwise XOR operation.

Return *the****maximum****possible bitwise XOR of all elements of*nums*after applying the operation****any number****of times*.

**Example 1:**

**Input:** nums = [3,2,4,6]

**Output:** 7

**Explanation:** Apply the operation with x = 4 and i = 3, num[3] = 6 AND (6 XOR 4) = 6 AND 2 = 2.

Now, nums = [3, 2, 4, 2] and the bitwise XOR of all the elements = 3 XOR 2 XOR 4 XOR 2 = 7.

It can be shown that 7 is the maximum possible bitwise XOR.

Note that other operations may be used to achieve a bitwise XOR of 7.

**Example 2:**

**Input:** nums = [1,2,3,9,2]

**Output:** 11

**Explanation:** Apply the operation zero times.

The bitwise XOR of all the elements = 1 XOR 2 XOR 3 XOR 9 XOR 2 = 11.

It can be shown that 11 is the maximum possible bitwise XOR.

**Constraints:**

* 1 <= nums.length <= 105
* 0 <= nums[i] <= 108