# Problem 2293: Min Max Game

## Problem Information

**Difficulty:** Easy
**Acceptance Rate:** 64.05%
**Paid Only:** No
**Tags:** Array, Simulation

## Problem Description

You are given a **0-indexed** integer array `nums` whose length is a power of `2`.

Apply the following algorithm on `nums`:

1. Let `n` be the length of `nums`. If `n == 1`, **end** the process. Otherwise, **create** a new **0-indexed** integer array `newNums` of length `n / 2`. 2. For every **even** index `i` where `0 <= i < n / 2`, **assign** the value of `newNums[i]` as `min(nums[2 * i], nums[2 * i + 1])`. 3. For every **odd** index `i` where `0 <= i < n / 2`, **assign** the value of `newNums[i]` as `max(nums[2 * i], nums[2 * i + 1])`. 4. **Replace** the array `nums` with `newNums`. 5. **Repeat** the entire process starting from step 1.

Return _the last number that remains in_`nums` _after applying the algorithm._

**Example 1:**

![](https://assets.leetcode.com/uploads/2022/04/13/example1drawio-1.png)

**Input:** nums = [1,3,5,2,4,8,2,2] **Output:** 1 **Explanation:** The following arrays are the results of applying the algorithm repeatedly. First: nums = [1,5,4,2] Second: nums = [1,4] Third: nums = [1] 1 is the last remaining number, so we return 1.

**Example 2:**

**Input:** nums = [3] **Output:** 3 **Explanation:** 3 is already the last remaining number, so we return 3.

**Constraints:**

* `1 <= nums.length <= 1024` * `1 <= nums[i] <= 109` * `nums.length` is a power of `2`.

## Code Snippets

### C++:

```
class Solution {
public:
int minMaxGame(vector<int>& nums) {

}
};
```

### Java:

```
class Solution {
public int minMaxGame(int[] nums) {

}
}
```

### Python3:

```
class Solution:
def minMaxGame(self, nums: List[int]) -> int:
```