

# Problem 1049: Last Stone Weight II

## Problem Information

**Difficulty:** Medium

**Acceptance Rate:** 58.71%

**Paid Only:** No

**Tags:** Array, Dynamic Programming

## Problem Description

You are given an array of integers `stones` where `stones[i]` is the weight of the `ith` stone.

We are playing a game with the stones. On each turn, we choose any two stones and smash them together. Suppose the stones have weights `x` and `y` with `x <= y`. The result of this smash is:

\* If `x == y`, both stones are destroyed, and \* If `x != y`, the stone of weight `x` is destroyed, and the stone of weight `y` has new weight `y - x` .

At the end of the game, there is **at most one** stone left.

Return **\_the smallest possible weight of the left stone\_**. If there are no stones left, return `0`.

**Example 1:**

**Input:** stones = [2,7,4,1,8,1] **Output:** 1 **Explanation:** We can combine 2 and 4 to get 2, so the array converts to [2,7,1,8,1] then, we can combine 7 and 8 to get 1, so the array converts to [2,1,1,1] then, we can combine 2 and 1 to get 1, so the array converts to [1,1,1] then, we can combine 1 and 1 to get 0, so the array converts to [1], then that's the optimal value.

**Example 2:**

**Input:** stones = [31,26,33,21,40] **Output:** 5

**Constraints:**

```
* `1 <= stones.length <= 30` * `1 <= stones[i] <= 100`
```

## Code Snippets

### C++:

```
class Solution {
public:
    int lastStoneWeightII(vector<int>& stones) {

    }
};
```

### Java:

```
class Solution {
    public int lastStoneWeightII(int[] stones) {

    }
}
```

### Python3:

```
class Solution:
    def lastStoneWeightII(self, stones: List[int]) -> int:
```