Problem 2335: Minimum Amount of Time to Fill Cups

Problem Information

Difficulty: Easy
Acceptance Rate: 59.66%
Paid Only: No
Tags: Array, Greedy, Sorting, Heap (Priority Queue)

Problem Description

You have a water dispenser that can dispense cold, warm, and hot water. Every second, you can either fill up `2` cups with **different** types of water, or `1` cup of any type of water.

You are given a **0-indexed** integer array `amount` of length `3` where `amount[0]`, `amount[1]`, and `amount[2]` denote the number of cold, warm, and hot water cups you need to fill respectively. Return _the**minimum** number of seconds needed to fill up all the cups_.

Example 1:

Input: amount = [1,4,2] **Output:** 4 **Explanation:** One way to fill up the cups is: Second 1: Fill up a cold cup and a warm cup. Second 2: Fill up a warm cup and a hot cup. Second 3: Fill up a warm cup and a hot cup. Second 4: Fill up a warm cup. It can be proven that 4 is the minimum number of seconds needed.

Example 2:

Input: amount = [5,4,4] **Output:** 7 **Explanation:** One way to fill up the cups is: Second 1: Fill up a cold cup, and a hot cup. Second 2: Fill up a cold cup, and a warm cup. Second 3: Fill up a cold cup, and a warm cup. Second 4: Fill up a warm cup, and a hot cup. Second 5: Fill up a cold cup, and a hot cup. Second 6: Fill up a cold cup, and a warm cup. Second 7: Fill up a hot cup.

Example 3:

Input: amount = [5,0,0] **Output:** 5 **Explanation:** Every second, we fill up a cold cup.

Constraints:

* `amount.length == 3` * `0 <= amount[i] <= 100`

Code Snippets

C++:

```
class Solution {
public:
int fillCups(vector<int>& amount) {

}
};
```

Java:

```
class Solution {
public int fillCups(int[] amount) {

}
}
```

Python3:

```
class Solution:
def fillCups(self, amount: List[int]) -> int:
```