Intuition

When I first read this problem, I realized that it's not just about using each battery to its full capacity before moving on to the next. The goal here is to maximize the running time of all computers, and a simple linear approach would not achieve that. The challenge here lies in striking a balance between using batteries that have the most power and ensuring that no energy is wasted.

The maximum possible ans can be sum of all batteries divide by no computers.
The minimum possible ans can 1.
so we found a range for ans from 1 to sum/n and we need maximum time to run computer => **BINARY SEARCH**

Approach

The approach I took for this problem is to find a balance between using batteries that have the most power and ensuring that no energy is wasted. I employed a binary search strategy to locate the maximum possible running time.

Initially, I set 1 as the left boundary and the total power of all batteries divided by 'n' as the right boundary. In the binary search, I set a target running time and checked if we could reach this target with the available batteries. If we could, I updated the left boundary to the target. If we couldn't, I updated the right boundary to one less than the target. I continued this process until the left and right boundaries met, at which point we've found the maximum possible running time.

- we will check is it possible to sustain n computers for 'hrs' amount of time for given configuration of batteries.
- 'hrs' for which we can run current configuration of batteries will depend on battery which runs for least amount of time.

Complexity

- Time complexity: The time complexity for this problem is (O(m log k)), where (m) is the length of the input array batteries and (k) is the maximum power of one battery.
- Space complexity: The space complexity for this problem is (O(1)). During the binary search, we only need to record the boundaries of the searching space and the power extra, and the accumulative sum of extra, which only takes constant space.