<https://leetcode.com/problems/capacity-to-ship-packages-within-d-days/description/>

A conveyor belt has packages that must be shipped from one port to another within days days.

The ith package on the conveyor belt has a weight of weights[i]. Each day, we load the ship with packages on the conveyor belt (in the order given by weights). We may not load more weight than the maximum weight capacity of the ship.

Return the least weight capacity of the ship that will result in all the packages on the conveyor belt being shipped within days days.

**Example 1:**

**Input:** weights = [1,2,3,4,5,6,7,8,9,10], days = 5

**Output:** 15

**Explanation:** A ship capacity of 15 is the minimum to ship all the packages in 5 days like this:

1st day: 1, 2, 3, 4, 5

2nd day: 6, 7

3rd day: 8

4th day: 9

5th day: 10

Note that the cargo must be shipped in the order given, so using a ship of capacity 14 and splitting the packages into parts like (2, 3, 4, 5), (1, 6, 7), (8), (9), (10) is not allowed.

**Example 2:**

**Input:** weights = [3,2,2,4,1,4], days = 3

**Output:** 6

**Explanation:** A ship capacity of 6 is the minimum to ship all the packages in 3 days like this:

1st day: 3, 2

2nd day: 2, 4

3rd day: 1, 4

**Example 3:**

**Input:** weights = [1,2,3,1,1], days = 4

**Output:** 3

**Explanation:**

1st day: 1

2nd day: 2

3rd day: 3

4th day: 1, 1

**Constraints:**

1 <= days <= weights.length <= 5 \* 10^4

1 <= weights[i] <= 500

**Attempt 1: 2024-12-06**

**Solution 1: Binary Search + Greedy (10 min, same as L410.Split Array Largest Sum)**

class Solution {

    public int shipWithinDays(int[] weights, int days) {

        int max = 0;

        int sum = 0;

        for(int w : weights) {

            max = Math.max(max, w);

            sum += w;

        }

        int lo = max;

        int hi = sum;

        while(lo <= hi) {

            int mid = lo + (hi - lo) / 2;

            if(findMinShipCapacity(weights, days, mid)) {

                hi = mid - 1;

            } else {

                lo = mid + 1;

            }

        }

        return lo;

    }

    private boolean findMinShipCapacity(int[] weights, int days, int minShipCapacity) {

        int requireDays = 1;

        long sum = 0;

        for(int w : weights) {

            sum += w;

            if(sum > minShipCapacity) {

                requireDays++;

                sum = w;

                // If required days exceed determined days, return false

                if(requireDays > days) {

                    return false;

                }

            }

        }

        return true;

    }

}

**Refer to**

[L410.Split Array Largest Sum (Ref.L1011,L704)](note://WEB4d9d6f26a965381b137615128e2648be)