Given an array of integers `nums` and an integer `threshold`, we will choose a p

Given an array of integers `nums` and an integer `threshold`, we will choose a positive integer `divisor`, divide all the array by it, and sum the division's result. Find the **smallest** `divisor` such that the result mentioned above is less than or equal to `threshold`.

Each result of the division is rounded to the nearest integer greater than or equal to that element. (For example: `7/3 = 3` and `10/2 = 5`).

It is guaranteed that there will be an answer.

Example 1:

Input: nums = [1,2,5,9], threshold = 6
Output: 5
Explanation: We can get a sum to 17 (1+2+5+9) if the divisor is 1.
If the divisor is 4 we can get a sum of 7 (1+1+2+3)
and if the divisor is 5 the sum will be 5 (1+1+1+2).

Example 2:

Input: nums = [44,22,33,11,1], threshold = 5
Output: 44

Example 3:

Input: nums = [21212,10101,12121], threshold = 1000000
Output: 1

Example 4:

Input: nums = [2,3,5,7,11], threshold = 11
Output: 3

```
def smallestDivisor(self, nums: List[int], threshold: int) -> int:
        left = 1
        right = max(nums)
        ans = -1
        while left<=right:
            mid = (left+right)//2
```

```
            if self.is_valid(nums,threshold,mid):
                ans = mid
                right = mid-1
            else:
                left = mid+1
        return ans


    def is_valid(self,nums,threshold,mid):
        sum = 0
        for ele in nums:
            temp = ele%mid
            if temp!=0:
                sum = sum+(ele//mid)+1
            else:
                sum = sum+(ele//mid)
        if sum>threshold:
            return False
        return True
```