**Main idea:**

**Increasing n - 1 smaller elements by 1 = decreasing only the max element by 1**

[1,2,3]

[2,3,3]

[3,4,3]

[4,4,4]

Else:

Min = 1

For each element in array:

Find its distance from the min

Return sum of distances

[1,2,3]

1 is 0 steps away

2 is 1 step away

3 is 2 steps away

= 0+1+2 = 3

Code:

def minMoves(self, nums: List[int]) -> int:

minEle = min(nums)

ans= 0

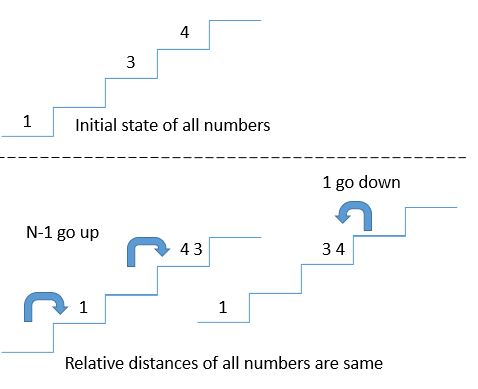
for num in nums:

ans+= num-minEle

return ans

—-----------------------------------------------------------------------------------------------------------------------

There is a staircase on which every numbers in array stand with corresponding step. '1' is on the 1st step and '5' on the 5th step. A single move makes n-1 numbers step up, while on the other hand, we can also think a move as the remaining one step down. The relative distance between the numbers are same.



**Approach in reverse manner:**

Our goal is to make all numbers on the same step.

Rather than move n-1 numbers up every time, why not just move one number down?

Solution:

1. Find min
2. Decrement all numbers to min

Equation:

***number of moves = nums[0]-min + nums[1]-min + .... +nums[n]-min = sum - n \* min***