# 1959. Minimum Total Space Wasted With K Resizing Operations

## Description

You are currently designing a dynamic array. You are given a **0-indexed** integer array `nums`, where `nums[i]` is the number of elements that will be in the array at time `i`. In addition, you are given an integer `k`, the **maximum** number of times you can **resize** the array (to **any** size).

The size of the array at time `t`, `size_t`, must be at least `nums[t]` because there needs to be enough space in the array to hold all the elements. The **space wasted** at time `t` is defined as `size_t - nums[t]`, and the **total** space wasted is the **sum** of the space wasted across every time `t` where `0 <= t < nums.length`.

Return *the **minimum** **total space wasted** if you can resize the array at most `k` times*.

**Note:** The array can have **any size** at the start and does **not** count towards the number of resizing operations.

### Example 1:

```
Input: nums = [10,20], k = 0
Output: 10
Explanation: size = [20,20].
We can set the initial size to be 20.
The total wasted space is (20 - 10) + (20 - 20) = 10.
```

### Example 2:

```
Input: nums = [10,20,30], k = 1
Output: 10
Explanation: size = [20,20,30].
We can set the initial size to be 20 and resize to 30 at time 2.
The total wasted space is (20 - 10) + (20 - 20) + (30 - 30) = 10.
```

### Example 3:

```
Input: nums = [10,20,15,30,20], k = 2
Output: 15
Explanation: size = [10,20,20,30,30].
We can set the initial size to 10, resize to 20 at time 1, and resize to 30 at time 3.
The total wasted space is (10 - 10) + (20 - 20) + (20 - 15) + (30 - 30) + (30 - 20) = 15.
```

### Constraints:

- `1 <= nums.length <= 200`
- `1 <= nums[i] <= 10^6`
- `0 <= k <= nums.length - 1`