You are given a **0-indexed** integer array piles, where piles[i] represents the number of stones in the ith pile, and an integer k. You should apply the following operation **exactly** k times:

* Choose any piles[i] and **remove** floor(piles[i] / 2) stones from it.

**Notice** that you can apply the operation on the **same** pile more than once.

Return *the****minimum****possible total number of stones remaining after applying the*k*operations*.

floor(x) is the **greatest** integer that is **smaller** than or **equal** to x (i.e., rounds x down).

**Example 1:**

**Input:** piles = [5,4,9], k = 2

**Output:** 12

**Explanation:** Steps of a possible scenario are:

- Apply the operation on pile 2. The resulting piles are [5,4,5].

- Apply the operation on pile 0. The resulting piles are [3,4,5].

The total number of stones in [3,4,5] is 12.

**Example 2:**

**Input:** piles = [4,3,6,7], k = 3

**Output:** 12

**Explanation:** Steps of a possible scenario are:

- Apply the operation on pile 3. The resulting piles are [4,3,3,7].

- Apply the operation on pile 4. The resulting piles are [4,3,3,4].

- Apply the operation on pile 0. The resulting piles are [2,3,3,4].

The total number of stones in [2,3,3,4] is 12.

**Constraints:**

* 1 <= piles.length <= 105
* 1 <= piles[i] <= 104
* 1 <= k <= 105