Problem 3296: Minimum Number of Seconds to Make Mountain Height Zero

Problem Information

Difficulty: Medium
Acceptance Rate: 36.52%
Paid Only: No
Tags: Array, Math, Binary Search, Greedy, Heap (Priority Queue)

Problem Description

You are given an integer `mountainHeight` denoting the height of a mountain.

You are also given an integer array `workerTimes` representing the work time of workers in **seconds**.

The workers work **simultaneously** to **reduce** the height of the mountain. For worker `i`:

* To decrease the mountain's height by `x`, it takes `workerTimes[i] + workerTimes[i] * 2 + ... + workerTimes[i] * x` seconds. For example: * To reduce the height of the mountain by 1, it takes `workerTimes[i]` seconds. * To reduce the height of the mountain by 2, it takes `workerTimes[i] + workerTimes[i] * 2` seconds, and so on.

Return an integer representing the **minimum** number of seconds required for the workers to make the height of the mountain 0.

Example 1:

Input: mountainHeight = 4, workerTimes = [2,1,1]

Output: 3

Explanation:

One way the height of the mountain can be reduced to 0 is:

* Worker 0 reduces the height by 1, taking `workerTimes[0] = 2` seconds. * Worker 1 reduces the height by 2, taking `workerTimes[1] + workerTimes[1] * 2 = 3` seconds. * Worker 2 reduces the height by 1, taking `workerTimes[2] = 1` second.

Since they work simultaneously, the minimum time needed is `max(2, 3, 1) = 3` seconds.

Example 2:

Input: mountainHeight = 10, workerTimes = [3,2,2,4]

Output: 12

Explanation:

* Worker 0 reduces the height by 2, taking `workerTimes[0] + workerTimes[0] * 2 = 9` seconds. * Worker 1 reduces the height by 3, taking `workerTimes[1] + workerTimes[1] * 2 + workerTimes[1] * 3 = 12` seconds. * Worker 2 reduces the height by 3, taking `workerTimes[2] + workerTimes[2] * 2 + workerTimes[2] * 3 = 12` seconds. * Worker 3 reduces the height by 2, taking `workerTimes[3] + workerTimes[3] * 2 = 12` seconds.

The number of seconds needed is `max(9, 12, 12, 12) = 12` seconds.

Example 3:

Input: mountainHeight = 5, workerTimes = [1]

Output: 15

Explanation:

There is only one worker in this example, so the answer is `workerTimes[0] + workerTimes[0] * 2 + workerTimes[0] * 3 + workerTimes[0] * 4 + workerTimes[0] * 5 = 15`.

Constraints:

* `1 <= mountainHeight <= 105` * `1 <= workerTimes.length <= 104` * `1 <= workerTimes[i] <= 106`

Code Snippets

C++:

```
class Solution {
public:
long long minNumberOfSeconds(int mountainHeight, vector<int>& workerTimes) {

}
};
```

Java:

```
class Solution {
public long minNumberOfSeconds(int mountainHeight, int[] workerTimes) {

}
}
```

Python3:

```
class Solution:
def minNumberOfSeconds(self, mountainHeight: int, workerTimes: List[int]) ->
int:
```