Problem 2015: Average Height of Buildings in Each Segment

Problem Information

Difficulty: Medium
Acceptance Rate: 58.02%
Paid Only: Yes
Tags: Array, Greedy, Sorting, Heap (Priority Queue)

Problem Description

A perfectly straight street is represented by a number line. The street has building(s) on it and is represented by a 2D integer array `buildings`, where `buildings[i] = [starti, endi, heighti]`. This means that there is a building with `heighti` in the **half-closed segment** `[starti, endi)`.

You want to **describe** the heights of the buildings on the street with the **minimum** number of non-overlapping **segments**. The street can be represented by the 2D integer array `street` where `street[j] = [leftj, rightj, averagej]` describes a **half-closed segment** `[leftj, rightj)` of the road where the **average** heights of the buildings in the**segment** is `averagej`.

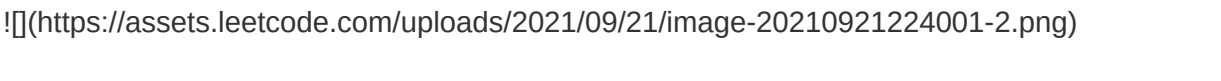
* For example, if `buildings = [[1,5,2],[3,10,4]],` the street could be represented by `street = [[1,3,2],[3,5,3],[5,10,4]]` because: * From 1 to 3, there is only the first building with an average height of `2 / 1 = 2`. * From 3 to 5, both the first and the second building are there with an average height of `(2+4) / 2 = 3`. * From 5 to 10, there is only the second building with an average height of `4 / 1 = 4`.

Given `buildings`, return _the 2D integer array_`street` _as described above (**excluding** any areas of the street where there are no buldings). You may return the array in **any order**_.

The **average** of `n` elements is the **sum** of the `n` elements divided (**integer division**) by `n`.

A **half-closed segment** `[a, b)` is the section of the number line between points `a` and `b` **including** point `a` and **not including** point `b`.

Example 1:



Input: buildings = [[1,4,2],[3,9,4]] **Output:** [[1,3,2],[3,4,3],[4,9,4]] **Explanation:** From 1 to 3, there is only the first building with an average height of 2 / 1 = 2. From 3 to 4, both the first and the second building are there with an average height of (2+4) / 2 = 3. From 4 to 9, there is only the second building with an average height of 4 / 1 = 4.

Example 2:

Input: buildings = [[1,3,2],[2,5,3],[2,8,3]] **Output:** [[1,3,2],[3,8,3]] **Explanation:** From 1 to 2, there is only the first building with an average height of 2 / 1 = 2. From 2 to 3, all three buildings are there with an average height of (2+3+3) / 3 = 2. From 3 to 5, both the second and the third building are there with an average height of (3+3) / 2 = 3. From 5 to 8, there is only the last building with an average height of 3 / 1 = 3. The average height from 1 to 3 is the same so we can group them into one segment. The average height from 3 to 8 is the same so we can group them into one segment.

Example 3:

Input: buildings = [[1,2,1],[5,6,1]] **Output:** [[1,2,1],[5,6,1]] **Explanation:** From 1 to 2, there is only the first building with an average height of 1 / 1 = 1. From 2 to 5, there are no buildings, so it is not included in the output. From 5 to 6, there is only the second building with an average height of 1 / 1 = 1. We cannot group the segments together because an empty space with no buildings seperates the segments.

Constraints:

* `1 <= buildings.length <= 105` * `buildings[i].length == 3` * `0 <= starti < endi <= 108` * `1 <= heighti <= 105`

Code Snippets

C++:

```
class Solution {
public:
vector<vector<int>> averageHeightOfBuildings(vector<vector<int>>& buildings)
```

```
{

}
};
```

Java:

```
class Solution {
public int[][] averageHeightOfBuildings(int[][] buildings) {

}
}
```

Python3:

```
class Solution:
def averageHeightOfBuildings(self, buildings: List[List[int]]) ->
List[List[int]]:
```