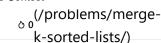
☐ [X Premium]
Store
(/subscribe?
ref=nb_npl)



User Accepted:



462

6318. Minimum Time to Complete All Tasks

My Submissions (/contest/weekly-contest-336/problems/minimum-time-to-complete-all-tasks/submissions/)

Back
There is a computer that can run an unlimited number of tasks at the same time. You are given a 2D integer array tasks

Back to Contest (/contest/weekly-contest-336/)

where tasks[i] = [start_i, end_i, duration_i] indicates that the ith task should run for a total of duration_i seconds (not necessarily continuous) within the **inclusive** time range [start_i, end_i].

User Tried: 1738

You may turn on the computer only when it needs to run a task. You can also turn it off if it is idle.

Total Accepted: 537

Return the minimum time during which the computer should be turned on to complete all tasks.

Total Submissions: 4162

Difficulty: (Hard)

Example 1:

```
Input: tasks = [[2,3,1],[4,5,1],[1,5,2]]
Output: 2
Explanation:
- The first task can be run in the inclusive time range [2, 2].
- The second task can be run in the inclusive time range [5, 5].
- The third task can be run in the two inclusive time ranges [2, 2] and [5, 5].
The computer will be on for a total of 2 seconds.
```

Example 2:

```
Input: tasks = [[1,3,2],[2,5,3],[5,6,2]]
Output: 4
Explanation:
- The first task can be run in the inclusive time range [2, 3].
- The second task can be run in the inclusive time ranges [2, 3] and [5, 5].
- The third task can be run in the two inclusive time range [5, 6].
The computer will be on for a total of 4 seconds.
```

Constraints:

```
• 1 <= tasks.length <= 2000
```

- tasks[i].length == 3
- 1 <= start_i, end_i <= 2000
- 1 <= $duration_i$ <= end_i $start_i$ + 1

```
Python
                                                                                                                                 ψ
                                                                                                                                       \mathfrak{C}
 1 ▼ class Solution(object):
 2 •
        def findMinimumTime(self, tasks):
 3
 4
             :type tasks: List[List[int]]
 5
             :rtype: int
 6
 7
             start = 100000
 8
             for i in range(len(tasks)):
 9 •
10 +
                 if tasks[i][0] < start:</pre>
11
                     start = tasks[i][0]
                 if tasks[i][1] > end:
12 •
13
                     end = tasks[i][1]
14
             countInstance = []
15
16
             countInstance2 = []
17 •
             for i in range(start, end+1, 1):
18
                 countInstance.append(0)
19
                 countInstance2.append(0)
20
             #print(countInstance)
21
22
23 •
             for i in range(len(tasks)):
24 ▼
                 for j in range(tasks[i][0], tasks[i][1]+1, 1):
25
                      countInstance[j-1] += 1
```

```
27 ▼
               for i in range(len(tasks)):
                    for j in range(tasks[i][0], tasks[i][1]+1, 1):
 28
 29
 30
               val = max(countInstance)
 31
 32
               print(val)
 33
               print(countInstance.count(val))
               print(countInstance.count(val)/2)
 34
 35
               return countInstance.count(val)/2 # I give up
 36
 37
 38
 39
 40
 41
 42
☐ Custom Testcase
                       Use Example Testcases
                                                                                                                                    Run
                                                                                                                                               △ Submit
Copyright © 2023 LeetCode
Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)
United States (/region)
```