< Back

# [Java/Python 3] simple code using PriorityQueue/heapq w/ brief explanation and analysis

37

rock ★ 5275 Last Edit: October 24, 2019 12:25 PM 2.2K VIEWS

1. Put both `slots1` and `slots2` into PriorityQueue/heapq (first filter slots shorter than `duration`, credit to **@SunnyvaleCA**), sort by starting time;
2. Pop out the slots one by one, comparing every consective two to check if having `duration` time in common.

```
    public List<Integer> minAvailableDuration(int[][] slots1, int[][] slots2, int duration) {
        PriorityQueue<int[]> pq = new PriorityQueue<>(Comparator.comparing(a -> a[0]));
        for (int[] s : slots1)
            if (s[1] - s[0] >= duration)
                pq.offer(s);
        for (int[] s : slots2)
            if (s[1] - s[0] >= duration)
                pq.offer(s);
        while (pq.size() > 1)
            if (pq.poll()[1] >= pq.peek()[0] + duration)
                return Arrays.asList(pq.peek()[0], pq.peek()[0] + duration);
        return Arrays.asList();
    }
```

```
    def minAvailableDuration(self, slots1: List[List[int]], slots2: List[List[int]], duration: int) -> List[int]:
        s = list(filter(lambda slot: slot[1] - slot[0] >= duration, slots1 + slots2))
        heapq.heapify(s)
        while len(s) > 1:
            if heapq.heappop(s)[1] >= s[0][0] + duration:
                return [s[0][0], s[0][0] + duration]
        return []
```

**Analysis**

Time: O(nlog(n)), space: O(n), where n = slots1.length + slots2.length.

Comments: 10 Best | Most Votes | Newest to Oldest | Oldest to Newest

Type comment here... (Markdown is supported)

Post

SunnyvaleCA ★ 262 October 20, 2019 1:49 AM

You can make things faster by filtering first...

```
    slots1 = list(filter(lambda slot: slot[1]-slot[0] >= duration, slots1))
    slots2 = ...
```

4 Show 3 replies Reply

zhuangr ★ 14 March 19, 2020 7:18 AM

Don't know if I understand the question correctly. So the slots1 is for person1, and slots2 is for person2 right? How to make sure in your solution that the poll element from priorityqueue and the peek element from priorityqueue are from different persons?
Thank you

wdj0xda ★ 293 February 12, 2020 10:15 PM

人狠话不多

0 Reply

leetspeak ★ 7 May 5, 2020 5:16 PM

Similar Approach. Here is my Java version

```
public List<Integer> minAvailableDuration(int[][] slots1, int[][] slots2, int duration) {
        List<Integer> res = new ArrayList<>();
        PriorityQueue<int[]> pq = new PriorityQueue<>((a, b) -> a[0] == b[0] ? a[1] - b[1] : a[0] - b[0]);
        for (int slot[] : slots1)
            if (slot[1] - slot[0] >= duration)
                pq.offer(slot);
        for (int slot[] : slots2)
            if (slot[1] - slot[0] >= duration)
                pq.offer(slot);
```

Read More

0 Reply

henry26 ★ 44 April 18, 2020 1:31 AM

rock, i see an error when i submit your solution.

0 Reply

th015 ★ 199 January 3, 2020 7:17 AM

Thanks for sharing.
It's line sweep algorithm implemented by heap.

0 Reply Share Report

totsubo ★ 618 December 20, 2019 1:08 PM

Super solution, thanks for sharing!

0 Reply

tlj77 ★ 273 December 2, 2019 5:32 AM

This is a really cool solution.

0 Reply

shuuchen ★ 142 Last Edit: October 29, 2019 8:42 PM

concise !
is it necessary to keep a heap ? It is time consuming.

```
class Solution:
    def minAvailableDuration(self, S1: List[List[int]], S2: List[List[int]], duration: int) -> List[int]:

        s = list(filter(lambda x: x[1] - x[0] >= duration, S1 + S2))
        s = sorted(s, reverse=True)

        while len(s) > 1:
            if s.pop()[1] >= s[-1][0] + duration:
```

Read More

0 Reply

mcclay ★ 79 October 20, 2019 2:51 AM

I dont really understand the heapq in general.

how is it different then doing the same thing without heapq?

```
class Solution():
    def minAvailableDuration(self, s1, s2, d):
        # [[Int]] [[Int]] Int -> [Int]

        s = sorted(s1 + s2)
        cur = s.pop(0)
        while s:
```

Read More

0 Show 1 reply Reply