AlgoExpert   Quad Layout   Python   12px   Sublime   Monokai   00:00:00

Prompt   Scratchpad   Our Solution(s)   Video Explanation   Run Code

Solution 1

```python
# Copyright © 2020 AlgoExpert, LLC. All rights reserved.

# O(c1 + c2) time | O(c1 + c2) space - where c1 and c2 are the respective numbers of meetings in calendar1 and calendar2
def calendarMatching(calendar1, dailyBounds1, calendar2, dailyBounds2, meetingDuration):
    updatedCalendar1 = updateCalendar(calendar1, dailyBounds1)
    updatedCalendar2 = updateCalendar(calendar2, dailyBounds2)
    mergedCalendar = mergeCalendars(updatedCalendar1, updatedCalendar2)
    flattenedCalendar = flattenCalendar(mergedCalendar)
    return getMatchingAvailabilities(flattenedCalendar, meetingDuration)


def updateCalendar(calendar, dailyBounds):
    updatedCalendar = calendar[:]
    updatedCalendar.insert(0, ["0:00", dailyBounds[0]])
    updatedCalendar.append([dailyBounds[1], "23:59"])
    return list(map(lambda m: [timeToMinutes(m[0]), timeToMinutes(m[1])], updatedCalendar))


def mergeCalendars(calendar1, calendar2):
    merged = []
    i, j = 0, 0
    while i < len(calendar1) and j < len(calendar2):
        meeting1, meeting2 = calendar1[i], calendar2[j]
        if meeting1[0] < meeting2[0]:
            merged.append(meeting1)
            i += 1
        else:
            merged.append(meeting2)
            j += 1
    while i < len(calendar1):
        merged.append(calendar1[i])
        i += 1
    while j < len(calendar2):
        merged.append(calendar2[j])
        j += 1
    return merged


def flattenCalendar(calendar):
    flattened = [calendar[0][:]]
    for i in range(1, len(calendar)):
        currentMeeting = calendar[i]
        previousMeeting = flattened[-1]
        currentStart, currentEnd = currentMeeting
        previousStart, previousEnd = previousMeeting
        if previousEnd >= currentStart:
            newPreviousMeeting = [previousStart, max(previousEnd, currentEnd)]
            flattened[-1] = newPreviousMeeting
        else:
            flattened.append(currentMeeting[:])
    return flattened


def getMatchingAvailabilities(calendar, meetingDuration):
    matchingAvailabilities = []
    for i in range(1, len(calendar)):
        start = calendar[i - 1][1]
        end = calendar[i][0]
        availabilityDuration = end - start
        if availabilityDuration >= meetingDuration:
            matchingAvailabilities.append([start, end])
    return list(map(lambda m: [minutesToTime(m[0]), minutesToTime(m[1])], matchingAvailabilities))


def timeToMinutes(time):
    hours, minutes = list(map(int, time.split(":")))
    return hours * 60 + minutes


def minutesToTime(minutes):
    hours = minutes // 60
    mins = minutes % 60
    hoursString = str(hours)
    minutesString = "0" + str(mins) if mins < 10 else str(mins)
    return hoursString + ":" + minutesString
```