

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(c1 + c2) time | O(c1 + c2) space
5     func calendarMatching(_ calendar1: [[String]], _ dailyBounds1: [String], _ calendar2: [[String]], _ dailyBounds2: [String]) -> [String] {
6         let updatedCalendar1 = updateCalendar(calendar1, dailyBounds1)
7         let updatedCalendar2 = updateCalendar(calendar2, dailyBounds2)
8
9         let mergedCalendar = mergeCalendars(updatedCalendar1, updatedCalendar2)
10        let flattenedCalendar = flattenCalendar(mergedCalendar)
11
12        return getMatchingAvailabilities(flattenedCalendar, meetingDuration)
13    }
14
15    func updateCalendar(_ calendar: [[String]], _ dailyBounds: [String]) -> [String] {
16        let lowerBound = ["0:00", dailyBounds[0]]
17        let upperBound = [dailyBounds[1], "23:59"]
18        var updatedCalendar = [[String]]()
19
20        for calendarItem in calendar {
21            updatedCalendar.append(lowerBound)
22            updatedCalendar.append(contentsOf: calendarItem)
23            updatedCalendar.append(upperBound)
24        }
25
26        return updatedCalendar.map { $0.map { timeToMinutes($0) } }
27    }
28
29    func mergeCalendars(_ calendar1: [[Int]], _ calendar2: [[Int]]) -> [[Int]] {
30        var i = 0
31        var j = 0
32        var merged = [[Int]]()
33
34        while i < calendar1.count & j < calendar2.count {
35            let meeting1 = calendar1[i]
```

Solution 1

Solution 2

Solution 3

```
1 class Program {
2     func calendarMatching(_ calendar1: [[String]], _ dailyBounds1: [String], _ calendar2: [[String]], _ dailyBounds2: [String]) -> [String] {
3         // Write your code here.
4         return []
5     }
6 }
7
```

Our Tests

Custom Output

Submit Code

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(c1 + c2) time | O(c1 + c2) space
5     func calendarMatching(_ calendar1: [[String]], _ dailyBounds1: [String], _ calendar2: [[String]], _ dailyBounds2: [String]) -> [String] {
6         let updatedCalendar1 = updateCalendar(calendar1, dailyBounds1)
7         let updatedCalendar2 = updateCalendar(calendar2, dailyBounds2)
8
9         let mergedCalendar = mergeCalendars(updatedCalendar1, updatedCalendar2)
10        let flattenedCalendar = flattenCalendar(mergedCalendar)
11
12        return getMatchingAvailabilities(flattenedCalendar, meetingDuration)
13    }
14
15    func updateCalendar(_ calendar: [[String]], _ dailyBounds: [String]) -> [String] {
16        let lowerBound = ["0:00", dailyBounds[0]]
17        let upperBound = [dailyBounds[1], "23:59"]
18        var updatedCalendar = [[String]]()
19
20        for calendarItem in calendar {
21            updatedCalendar.append(lowerBound)
22            updatedCalendar.append(contentsOf: calendarItem)
23            updatedCalendar.append(upperBound)
24        }
25
26        return updatedCalendar.map { $0.map { timeToMinutes($0) } }
27    }
28
29    func mergeCalendars(_ calendar1: [[Int]], _ calendar2: [[Int]]) -> [[Int]] {
30        var i = 0
31        var j = 0
32        var merged = [[Int]]()
33
34        while i < calendar1.count & j < calendar2.count {
35            let meeting1 = calendar1[i]
```

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(c1 + c2) time | O(c1 + c2) space
5     func calendarMatching(_ calendar1: [[String]], _ dailyBounds1: [String], _ calendar2: [[String]], _ dailyBounds2: [String]) -> [String] {
6         // Write your code here.
7         return []
8     }
9 }
```

```

18     def test1():
19         program = Program()
20         method("Test Case 1", 1, 1, inputs = test1_in)
21         test_result = program.evaluateTesting(methods, variables)
22         test_result_expected, result
23     if
24         method("Test Case 2", 2, 1, inputs = test2_in)
25         constants = ("10000", "10000", "10000", "10000", "10000")
26         variables = ("10000", "10000")
27         constants = ("10000", "10000", "10000", "10000", "10000")
28         variables = ("10000", "10000")
29         string_literals = [""]
30         method = ("10000", "10000")
31         test_result = program.evaluateTesting(methods, variables)

```

Run or submit code when you're ready.