Solution 1

```
// 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
function calendarMatching(calendar1, dailyBounds1, calendar2, dailyBounds2, meetingDuration) {
  const updatedCalendar1 = updateCalendar(calendar1, dailyBounds1);
  const updatedCalendar2 = updateCalendar(calendar2, dailyBounds2);
  const mergedCalendar = mergeCalendars(updatedCalendar1, updatedCalendar2);
  const flattenedCalendar = flattenCalendar(mergedCalendar);
  return getMatchingAvailabilities(flattenedCalendar, meetingDuration);
}

function updateCalendar(calendar, dailyBounds) {
  const updatedCalendar = [['0:00', dailyBounds[0]], ...calendar, [dailyBounds[1], '23:59']];
  return updatedCalendar.map(meeting => meeting.map(timeToMinutes));
}

function mergeCalendars(calendar1, calendar2) {
  const merged = [];
  let i = 0,
    j = 0;
  while (i < calendar1.length && j < calendar2.length) {
    const meeting1 = calendar1[i],
      meeting2 = calendar2[j];
    if (meeting1[0] < meeting2[0]) {
      merged.push(meeting1);
      i++;
    } else {
      merged.push(meeting2);
      j++;
    }
  }
  while (i < calendar1.length) merged.push(calendar1[i++]);
  while (j < calendar2.length) merged.push(calendar2[j++]);
  return merged;
}

function flattenCalendar(calendar) {
  const flattened = [calendar[0].slice()];
  for (let i = 1; i < calendar.length; i++) {
    const currentMeeting = calendar[i];
    const previousMeeting = flattened[flattened.length - 1];
    const [currentStart, currentEnd] = currentMeeting;
    const [previousStart, previousEnd] = previousMeeting;
    if (previousEnd >= currentStart) {
      const newPreviousMeeting = [previousStart, Math.max(previousEnd, currentEnd)];
      flattened[flattened.length - 1] = newPreviousMeeting;
    } else {
      flattened.push(currentMeeting.slice());
    }
  }
  return flattened;
}

function getMatchingAvailabilities(calendar, meetingDuration) {
  const matchingAvailabilities = [];
  for (let i = 1; i < calendar.length; i++) {
    const start = calendar[i - 1][1];
    const end = calendar[i][0];
    const availabilityDuration = end - start;
    if (availabilityDuration >= meetingDuration) {
      matchingAvailabilities.push([start, end]);
    }
  }
  return matchingAvailabilities.map(meeting => meeting.map(minutesToTime));
}

function timeToMinutes(time) {
  const [hours, minutes] = time.split(':').map(str => parseInt(str));
  return hours * 60 + minutes;
}

function minutesToTime(minutes) {
  const hours = Math.floor(minutes / 60);
  const mins = minutes % 60;
  const hoursString = hours.toString();
  const minutesString = mins < 10 ? '0' + mins.toString() : mins.toString();
  return hoursString + ':' + minutesString;
}

exports.calendarMatching = calendarMatching;

```