File Header

```
#!/usr/bin/env node
// ชื่อ (ไม่ต้องใส่นามสกุล)
// รหัสนศ
// Sec00x
```

2) 100 คะแนน (GDR04_2_6XXXXXXXX.js) ท่ามกลางพายุฝนโหมกระหน่ำ ขบวนรถบัสจากบริษัทแดงและบริษัทส้ม สองบริษัทคู่แข่งที่บาดหมางกันมานานจากอุดมคติทางการเมืองที่แตกต่าง กำลังพานักท่องเที่ยวต่างชาติไปพักณ โรงแรมชื่อดังบนอุทยานแห่งชาติเขาใหญ่ที่เป็นข่าวดังเมื่อไม่นานมานี้ ทางขึ้นโรงแรมเป็นถนน One way สองเลน รถบัสทั้งหมดของบริษัทแดงวิ่งอยู่ทางเลนซ้าย รถบัสทั้งหมดของบริษัทส้มวิ่งอยู่ทางเลนขวา ทีมคนขับรถทั้งสองบริษัทต่างเหยียบคันเร่งสุดกำลังหวังจะไปถึงที่หมายก่อน โชคร้ายที่ก่อนถึงโรงแรมประมาณ 1 กิโลเมตร ทางเลนขวามีต้นไม้หักโค่นลงมาทำให้ต้องเดินรถได้เพียงทางเดียว เพื่อความปลอดภัย คนขับรถจากทั้งสองบริษัทจึงตกลงร่วมกันด้วย MOU ว่าจะไม่มีการหักหลังกัน จะไม่มีการขับถอยหลัง และรถที่มาจากเลนเดียวกันจะไม่แซงกัน

ให้เขียนฟังก์ชัน recursive `arrivalSequences(`*left_lane*`, `*right_lane*`)` เพื่อคืนค่า Array ของ String แทนลำดับที่เป็นไปได้ทั้งหมดที่รถบัสของทั้งสองบริษัทจะถึงโรงแรม โดย *left_lane* เป็น Array ของ String แทนเลขประจำรถจากบริษัทแดง และ *right_lane* เป็น Array ของ String แทนเลขประจำรถจากบริษัทส้ม กำหนดให้แต่ละบริษัทมีรถในขบวนอย่างน้อยหนึ่งคัน โดย String ของลำดับที่เป็น Output จะอยู่ในรูป เลขประจำรถคั่นด้วย '>' (เครื่องหมายมากกว่า) ทั้งนี้ String ใน Array ที่คืนค่าจะอยู่ในลำดับใดก็ได้ และให้ศึกษา template จากสัปดาห์ก่อนๆ ในการเขียนฟังก์ชันและจะต้องมีการ export ฟังก์ชัน `arrivalSequences()` อย่างถูกต้องเพื่อการเรียกใช้บนระบบตรวจให้คะแนน

Hint: สามารถเรียกใช้ method `assert.deepStrictEqual()` ในการเปรียบเทียบ Array เช่น

```
assert.deepStrictEqual(arrivalSequences(['R32'], ['09', '05']).sort(),
    ['09>05>R32', '09>R32>05', 'R32>09>05']);
```

Input	**Output**
`arrivalSequences(['R32'], ['09', '05']).sort()`	`[ '09>05>R32', '09>R32>05', 'R32>09>05' ]`
`arrivalSequences(['R2', 'R4'], ['034', '022']).sort()`	`[ '034>022>R2>R4', '034>R2>022>R4', '034>R2>R4>022', 'R2>034>022>R4', 'R2>034>R4>022', 'R2>R4>034>022' ]`