ตรวจสอบความปลอดภัยของโค้ด

ณ บริษัทผลิตรถยนต์ไฟฟ้าชั้นนำของโลก บริษัทของคุณกำลังพัฒนาโรงงานอัจฉริยะที่ใช้ระบบอัตโนมัติและปัญญาประดิษฐ์ในการควบคุมกระบวนการผลิตทั้งหมด

เนื่องจากความปลอดภัยเป็นสิ่งสำคัญสูงสุด ทีมของคุณได้รับมอบหมายให้พัฒนาระบบตรวจสอบความปลอดภัยของโค้ดที่ใช้ในการควบคุมหุ่นยนต์และเครื่องจักรในสายการผลิต โดยเฉพาะอย่างยิ่ง การตรวจสอบความถูกต้องของวงเล็บในสมการและเงื่อนไขต่างๆ ที่ใช้ในโค้ด

ข้อผิดพลาดในการจัดวางวงเล็บแม้เพียงเล็กน้อยอาจนำไปสู่การทำงานที่ผิดพลาดของหุ่นยนต์ ซึ่งอาจก่อให้เกิดอันตรายต่อพนักงานหรือความเสียหายต่อผลิตภัณฑ์ได้

หัวหน้าทีมความปลอดภัยของระบบมอบหมายให้คุณพัฒนาฟังก์ชันสำหรับตรวจสอบความถูกต้องของวงเล็บในโค้ดควบคุม โดยใช้โครงสร้างข้อมูล Stack เพื่อให้มั่นใจว่าไม่มีข้อผิดพลาดที่อาจนำไปสู่อุบัติเหตุในสายการผลิต

ข้อกำหนด

1. เขียนฟังก์ชัน check_brackets(code) ที่รับโค้ดควบคุมเป็น string
2. ฟังก์ชันต้องตรวจสอบเฉพาะวงเล็บเปิด "(" และวงเล็บปิด ")" เท่านั้น
3. ฟังก์ชันต้องคืนค่า True ถ้าวงเล็บปิดครบคู่ทุกอัน และ False ถ้าไม่ครบคู่
4. โค้ดอาจมีตัวเลข ตัวอักษร และสัญลักษณ์อื่นๆ แต่ไม่ต้องสนใจความถูกต้องทางไวยากรณ์ของโค้ด

ข้อมูลนำเข้า

- บรรทัดแรก: จำนวนเต็ม N แทนจำนวนโค้ดควบคุมที่ต้องตรวจสอบ ($1 \le N \le 100$)
- N บรรทัดถัดไป: แต่ละบรรทัดเป็นโค้ดควบคุมที่ต้องตรวจสอบ (ความยาวไม่เกิน 1000 ตัวอักษร)

ข้อมูลส่งออก

- N บรรทัด แต่ละบรรทัดแสดงผลการตรวจสอบเป็น "Safe" ถ้าวงเล็บปิดครบคู่ หรือ "Danger" ถ้าวงเล็บไม่ครบคู่

ตัวอย่างข้อมูลนำเข้า

INPUT	OUTPUT

5
if (sensor.getDistance() < SAFE_DISTANCE) { robot.stop(); }
while ((battery.level > 20) && (task.isNotComplete()) { robot.continue(); }
for (int i = 0; i < parts.length; i++) { assembly.add(parts[i]); }
if ((pressure > MAX_PRESSURE) || (temperature > MAX_TEMP) { emergency.shutdown(); }
switch (error.type) { case CRITICAL: alarm.activate(); break; default: log(error); }
	Safe
	Danger
	Safe
	Danger
	Safe

คำอธิบายตัวอย่าง

- โค้ดที่ 1, 3, และ 5 มีวงเล็บปิดครบคู่ จึงเป็น Safe
- โค้ดที่ 2 มีวงเล็บเปิดเกินมา 1 อัน จึงเป็น Danger
- โค้ดที่ 4 ขาดวงเล็บปิด 1 อัน จึงเป็น Danger