Badminton

เวลา: 1 วินาที | หน่วยความจำ: 256 MB

ในโรงเรียนมีคอร์ทแบตทั้งหมด 3 คอร์ท มีนักเรียนคนหนึ่งต้องการรู้ว่าคอร์ทหมายเลขใดว่างอยู่ ครูเวรฝั่งหน้าโรงยิมมองเห็นสถานะของคอร์ททั้ง 3 คอร์ท แต่ครูอีกคนที่อยู่หน้าห้องพักครูต้องเป็นคนตอบนักเรียน

สถานะของแต่ละคอร์ทเป็น 0 หรือ 1 โดย 0 หมายถึงคอร์ทว่าง และ 1 หมายถึงคอร์ทไม่ว่าง ครูเวรคนแรกต้องส่งข้อความเป็นจำนวนเต็มเพียง 1 จำนวนให้ครูคนที่สอง จากนั้นครูคนที่สองต้องตอบหมายเลขคอร์ทว่างที่มีหมายเลขน้อยที่สุด หากไม่มีคอร์ทว่างให้ตอบ 0

โจทย์นี้เป็นโจทย์แบบสื่อสาร ผู้เข้าแข่งขันต้องเขียนฟังก์ชัน 2 ฟังก์ชัน คือฟังก์ชันสำหรับผู้ส่งและฟังก์ชันสำหรับผู้รับ

รายละเอียดการเขียนโปรแกรม

ผู้เข้าแข่งขันต้องส่งไฟล์ `badminton.cpp` ที่ include ไฟล์ `badminton.h` และเขียนฟังก์ชันต่อไปนี้

```
int send_message(int court1, int court2, int court3);
int answer(int message);
```

ฟังก์ชัน `send_message` จะถูกเรียกโดยให้ค่า `court1`, `court2`, `court3` เป็นสถานะของคอร์ทหมายเลข $1, 2, 3$ ตามลำดับ ฟังก์ชันนี้ต้องคืนค่าข้อความเป็นจำนวนเต็มหนึ่งจำนวนในช่วง 0 ถึง 7

ฟังก์ชัน `answer` จะถูกเรียกโดยให้ค่า `message` เป็นค่าที่ฟังก์ชัน `send_message` คืนมา ฟังก์ชันนี้ต้องคืนค่าหมายเลขคอร์ทว่างที่มีหมายเลขน้อยที่สุด หรือคืนค่า 0 ถ้าไม่มีคอร์ทว่าง

ข้อมูลนำเข้า

สำหรับ grader ที่ให้มา ข้อมูลนำเข้ามีรูปแบบดังนี้

- บรรทัดที่ 1 มีจำนวนเต็ม q แทนจำนวนสถานการณ์ที่ต้องทดสอบ
- อีก q บรรทัด แต่ละบรรทัดมีจำนวนเต็ม 3 จำนวน คือ $court1, court2, court3$

ในการตัดสินจริง ผู้เข้าแข่งขันไม่ต้องอ่านข้อมูลนำเข้าเอง และไม่ต้องพิมพ์ข้อมูลส่งออกเอง

ข้อมูลส่งออก

สำหรับ grader ที่ให้มา ถ้าฟังก์ชันของผู้เข้าแข่งขันตอบถูกทุกสถานการณ์ grader จะพิมพ์ `Correct` มิฉะนั้นจะพิมพ์ `Wrong Answer`

ในการตัดสินจริง ผู้เข้าแข่งขันไม่ต้องพิมพ์ข้อมูลส่งออกเอง

ข้อกำหนด

- $1 \le q \le 1000$
- $court_i \in 0, 1$
- ค่าที่ `send_message` คืนมาต้องอยู่ในช่วง 0 ถึง 7

ตัวอย่างการเรียกฟังก์ชัน

ตัวอย่างต่อไปนี้แสดงลำดับการเรียกฟังก์ชันของ grader เท่านั้น ค่า `message` ที่ยกตัวอย่างไม่จำเป็นต้องตรงกับวิธีของผู้เข้าแข่งขันทุกคน ตราบใดที่ `answer` สามารถถอดรหัสข้อความที่ `send_message` ส่งมาได้ถูกต้อง

การเรียกฟังก์ชัน	คำอธิบาย
send_message(0, 1, 1) คืนค่า 6 answer(6) คืนค่า 1	คอร์ทที่ 1 ว่าง ส่วนคอร์ทที่ 2 และ 3 ไม่ว่าง ดังนั้นคำตอบที่ถูกต้องคือ 1
send_message(1, 0, 0) คืนค่า 1 answer(1) คืนค่า 2	คอร์ทที่ 1 ไม่ว่าง แต่คอร์ทที่ 2 และ 3 ว่าง คอร์ทว่างที่มีหมายเลขน้อยที่สุดคือคอร์ทที่ 2
send_message(1, 1, 1) คืนค่า 7 answer(7) คืนค่า 0	ทุกคอร์ทไม่ว่าง จึงต้องตอบ 0

คำอธิบายตัวอย่าง

ในแต่ละสถานการณ์ grader จะเรียก `send_message` ก่อน โดยส่งสถานะของคอร์ททั้ง 3 คอร์ทให้ฟังก์ชันนี้ จากนั้น grader จะนำค่าที่ `send_message` คืนมาไปเรียก `answer` อีกครั้ง ผู้รับจะเห็นเฉพาะค่า `message` เท่านั้น ไม่เห็นค่า `court1`, `court2`, `court3`

ตัวอย่างข้างต้นใช้วิธีเข้ารหัสแบบหนึ่งเท่านั้น เช่น `send_message(0, 1, 1)` คืนค่า `6` แต่ผู้เข้าแข่งขันสามารถเลือกส่งค่าอื่นได้ หากฟังก์ชัน `answer` ของตนเองแปลค่านั้นแล้วตอบ 1 ได้ถูกต้อง

ปัญหาย่อย

ปัญหาย่อย	ข้อกำหนดเพิ่มเติม	คะแนน

1	มีคอร์ทว่างอย่างน้อย 1 คอร์ทเสมอ	40
2	ไม่มีข้อกำหนดเพิ่มเติม	60