ข้อความ

ไอช่าและบาสม่าเป็นเพื่อนที่ติดต่อสื่อสารกันเป็นประจำ ไอช่ามีข้อความ M ที่เป็นลำดับของบิตความยาว S บิต (นั่นคือเป็นศูนย์และหนึ่ง) ที่เธอต้องการจะส่งให้กับบาสม่า ไอช่าสื่อสารกับบาสม่าโดยการส่ง **แพคเก็ต** (packet) แพคเก็ตหนึ่งคือลำดับของบิตความยาว 31 บิต ที่ระบุตำแหน่งเป็นค่าตั้งแต่ 0 ถึง 30 ไอช่าต้องการจะส่งข้อความ M ไปยังบาสม่า ผ่านทางการส่งแพคเก็ตจำนวนหนึ่ง

โชคไม่ดีที่คลีโอพัตราสามารถก่อกวนระบบสื่อสารระหว่างไอช่ากับบาสม่า ทำให้เธอสามารถ**บิดเบือน**แพคเก็ตเหล่านี้ได้ นั่นคือในแต่ละแพคเก็ต คลีโอพัตราสามารถแก้บิตในลำดับเป็นจำนวน 15 ตำแหน่ง กล่าวคือ จะมีอาร์เรย์ C ที่มีความยาว 31 โดยที่ข้อมูลแต่ละตัวในอาร์เรย์จะมีค่าเป็น 0 หรือ 1 ที่ระบุความหมายดังนี้:

- $C[i] = 1$ ระบุว่าคลีโอพัตราสามารถแก้ไขบิตที่ตำแหน่งที่ i ได้ เราจะเรียกตำแหน่งเหล่านี้ว่าเป็นตำแหน่งที่**ถูกควบคุม**โดยคลีโอพัตรา
- $C[i] = 0$ ระบุว่าคลีโอพัตราไม่สามารถแก้บิตที่ตำแหน่ง i ได้

ข้อมูลในอาร์เรย์ C จะต้องมีค่าที่เป็น 1 จำนวน 15 ค่าและมีค่าเป็น 0 จำนวน 16 ค่า ระหว่างที่ส่งข้อความ เซตของตำแหน่งที่ถูกควบคุมโดยคลีโอพัตราจะไม่มีการเปลี่ยนแปลงในทุก ๆ แพคเก็ต ไอช่าทราบว่าตำแหน่งจำนวน 15 ตำแหน่งที่ถูกควบคุมโดยคลีโอพัตราคือตำแหน่งใดบ้าง ส่วนบาสม่าทราบแค่ว่ามีตำแหน่งจำนวน 15 ตำแหน่งที่ถูกควบคุมเท่านั้น โดยไม่ทราบว่าเป็นตำแหน่งใดบ้าง

ให้ A เป็นแพคเก็ตที่ไอช่าตัดสินใจจะส่ง (ที่เราจะเรียกว่า **แพคเก็ตต้นฉบับ**) ให้ B เป็นแพคเก็ตที่บาสม่าได้รับ (ที่เราจะเรียกว่า **แพคเก็ตที่ถูกบิดเบือน**) สำหรับแต่ละตำแหน่ง i ที่ $0 \le i < 31$:

- ถ้าคลีโอพัตราไม่ได้ควบคุมบิตที่ตำแหน่ง i ($C[i] = 0$) บาสม่าจะได้รับบิตที่ตำแหน่ง i ตามที่ไอช่าส่งมา (นั่นคือ $B[i] = A[i]$)
- ไม่เช่นนั้น คลีโอพัตราจะควบคุมบิตที่ตำแหน่ง i ($C[i] = 1$) นั่นคือคลีโอพัตราจะสามารถกำหนดค่าของบิต $B[i]$ ได้

หลังจากที่ส่งแต่ละแพคเก็ตแล้ว ไอช่าจะได้รับแพคเก็ตที่ถูกบิดเบือนกลับมาพิจารณาโดยทันที

เมื่อไอช่าส่งแพคเก็ตทั้งหมดแล้ว บาสม่าจะได้รับแพคเก็ตที่ถูกบิดเบือนทั้งหมด **ตามลำดับที่แพคเก็ตเหล่านั้นถูกส่งมา** และจะต้องสร้างข้อความต้นฉบับ M กลับมา

งานของคุณคือการออกแบบและเขียนกลยุทธ์ที่ทำให้ไอช่าสามารถส่งข้อความ M ไปยังบาสม่า โดยที่บาสม่ายังสามารถสร้างข้อความต้นฉบับ M กลับมาได้จากแพคเก็ตที่ถูกบิดเบือน กล่าวคือ คุณจะต้องเขียนฟังก์ชันจำนวนสองฟังก์ชัน ฟังก์ชันแรกจะต้องทำงานในบทบาทของไอช่า นั่นคือฟังก์ชันจะรับข้อความ M และอาร์เรย์ C จากนั้นจะต้องส่งแพคเก็ตจำนวนหนึ่งเพื่อที่จะส่งข้อความไปยังบาสม่า ฟังก์ชันที่สองจะทำงานในบทบาทของบาสม่า นั่นคือฟังก์ชันจะได้รับแพคเก็ตที่ถูกบิดเบือนจากนั้นจะต้องสร้างข้อความต้นฉบับ M กลับมา

รายละเอียดการเขียนโปรแกรม

คุณจะต้องเขียนฟังก์ชันแรกในรูปแบบดังนี้:

```
void send_message(std::vector<bool> M, std::vector<bool> C)
```

- M: อาร์เรย์ความยาว S ที่ระบุ ข้อความที่ไอซ่าต้องการจะส่งให้กับบาสม่า
- C: อาร์เรย์ความยาว 31 ที่ระบุตำแหน่งของบิตที่ถูกควบคุมโดยคลีโอพัตรา
- ฟังก์ชันนี้จะถูกเรียก **ไม่เกิน 2100 ครั้ง** ในแต่ละกรณีทดสอบ

ฟังก์ชันนี้จะต้องเรียกใช้ฟังก์ชันด้านล่างเพื่อส่งแพคเก็ต:

```
std::vector<bool> send_packet(std::vector<bool> A)
```

- A: แพคเก็ตต้นฉบับ (เป็นอาร์เรย์ความยาว 31) ที่แทนลำดับของบิตที่ไอซ่าต้องการส่ง
- ฟังก์ชันนี้จะคืนแพคเก็ตที่ถูกบิดเบือน B ที่แทนลำดับของบิตที่บาสม่าจะได้รับ
- จะสามารถเรียกฟังก์ชันนี้ได้ไม่เกิน 100 ครั้ง ต่อการเรียกใช้ฟังก์ชัน `send_message` หนึ่งครั้ง

ฟังก์ชันที่สองที่คุณจะต้องเขียนอยู่ในรูปแบบ:

```
std::vector<bool> receive_message(std::vector<std::vector<bool>> R)
```

- R: อาร์เรย์ระบุแพคเก็ตที่ถูกบิดเบือนทั้งหมด แพคเก็ตดังกล่าวจะมาจากแพคเก็ตเริ่มต้นที่ไอซ่าส่งในการเรียกฟังก์ชัน `send_message` หนึ่งครั้ง และจะ**เรียงตามลำดับเดียวกันกับที่แพคเก็ตถูกส่ง**โดยไอซ่า ข้อมูลแต่ละตัวใน R จะเป็นอาร์เรย์ความยาว 31 แทนแพคเก็ตที่ถูกบิดเบือน
- ฟังก์ชันดังกล่าวจะคืนอาร์เรย์ความยาว S บิต ที่ตรงกับข้อความเริ่มต้น M
- ฟังก์ชันดังกล่าวอาจจะถูกเรียก**หลายครั้ง**ในแต่ละกรณีทดสอบ โดยจะมีการเรียกจำนวน**หนึ่งครั้งเท่านั้น**สำหรับแต่ละการเรียกฟังก์ชัน `send_message` **การเรียงลำดับของการเรียกฟังก์ชัน** `receive_message` ไม่จำเป็นต้องเรียงตามลำดับเดียวกันกับการเรียกฟังก์ชัน `send_message` ที่สอดคล้องกัน

สังเกตว่าในระบบเกรดเดอร์ ฟังก์ชัน `send_message` และฟังก์ชัน `receive_message` จะถูกเรียกโดย**โปรแกรมสองโปรแกรมที่แยกกัน**

เงื่อนไข

- $1 \leq S \leq 1024$
- อาร์เรย์ C จะมีขนาด 31 โดยที่ข้อมูล 16 ตัวจะมีค่าเท่ากับ 0 และข้อมูล 15 ตัวจะมีค่าเท่ากับ 1

ปัญหาย่อยและการให้คะแนน

ถ้าในกรณีทดสอบใด การเรียกฟังก์ชัน `send_packet` ไม่ตรงกับกฎที่ระบุไว้ข้างต้น หรือค่าที่คืนจากฟังก์ชัน `receive_message` ไม่ถูกต้อง คะแนนของคำตอบที่คุณได้รับสำหรับกรณีทดสอบนั้นจะเท่ากับ 0

ไม่เช่นนั้น ให้ Q แทนจำนวนครั้งที่มากที่สุดที่มีการเรียกใช้ฟังก์ชัน `send_packet` เมื่อพิจารณาจากทุก ๆ การเรียกฟังก์ชัน `send_message` ในกรณีทดสอบทั้งหมด นอกจากนั้น ให้ X มีค่าเท่ากับ:

- 1, ถ้า $Q \leq 66$
- 0.95^{Q-66}, ถ้า $66 < Q \leq 100$

คะแนนของคุณจะถูกคำนวณตามตารางต่อไปนี้:

ปัญหาย่อย	คะแนน	เงื่อนไขเพิ่มเติม
1	$10 \cdot X$	$S \leq 64$
2	$90 \cdot X$	ไม่มีเงื่อนไขเพิ่มเติม

สังเกตว่าในบางกรณีทดสอบ พฤติกรรมของเกรดเดอร์อาจจะ**ปรับเปลี่ยนได้** นั่นคือค่าที่คืนกลับมาในการเรียกฟังก์ชัน `send_packet` อาจจะไม่ได้ขึ้นกับข้อมูลนำเข้าของฟังก์ชันเท่านั้น แต่อาจขึ้นกับสิ่งอื่น ๆ ด้วย รวมถึงข้อมูลนำเข้า และค่าที่คืนจากการเรียกฟังก์ชันครั้งก่อน และจำนวนสุ่มเทียม (pseudo-random number) ที่สร้างจากเกรดเดอร์ เกรดเดอร์จะทำงานแบบมีการกำหนดไว้คงที่ (deterministic) ในความหมายที่ว่า ถ้าคุณเรียกใช้เกรดเดอร์สองครั้งและทั้งสองครั้งคุณส่งแพคเก็ตเหมือนกัน เกรดเดอร์จะบิดเบือนแพคเก็ตในลักษณะเดียวกันเสมอ

ตัวอย่าง

พิจารณาการเรียกฟังก์ชันดังนี้

```
send_message([0, 1, 1, 0],
             [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

ข้อความที่ไอช่าพยายามจะส่งให้บาสม่าคือ $[0, 1, 1, 0]$ คลีโอพัตราไม่สามารถบิดเบือนบิตที่ตำแหน่งตั้งแต่ 0 ถึง 15 ได้ ในขณะที่คลีโอพัตราสามารถบิดเบือนบิตที่ตำแหน่งตั้งแต่ 16 ถึง 30 ได้

สำหรับตัวอย่างนี้ สมมติว่าคลีโอพัตราจะใส่ค่าลงในบิตที่ติดกันที่เธอควบคุมได้ด้วยค่า 0 และ 1 สลับกัน นั่นคือเธอจะกำหนดให้บิตแรกที่เธอควบคุมเป็น 0 (ตำแหน่งที่ 16 ในกรณีนี้), ให้บิตที่สองที่เธอควบคุมเป็น 1 (ตำแหน่งที่ 17), ให้บิตที่สามที่เธอควบคุมเป็น 0 (ตำแหน่งที่ 18) เช่นนี้ไปเรื่อย ๆ

ไอช่าตัดสินใจที่จะส่งบิตสองบิตแรกจากข้อความตั้งต้นลงในแพคเก็ตเดียวดังนี้: เธอจะส่งบิตแรกลงใน 8 บิตแรกที่เธอควบคุม และบิตที่สองลงในบิตจำนวน 8 บิตถัดไปที่เธอควบคุม

ไอช่าจึงเลือกส่งแพคเก็ตต่อไปนี้:

```
send_packet([0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
             0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

สังเกตว่าคลีโอพัตราสามารถเปลี่ยนบิตในตำแหน่ง 15 ตำแหน่งท้าย ดังนั้นไอซ่าจึงกำหนดค่าลงไปเป็นอะไรก็ได้ เพราะว่าค่าเหล่านั้นอาจจะถูกบิดเบือนเป็นอะไรก็ได้ ด้วยกลยุทธ์ของคลีโอพัตราที่ระบุไว้ตอนแรก ฟังก์ชันจะคืนค่า: $[0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1]$.

ไอซ่าตัดสินใจที่จะส่งบิตสองบิตท้ายของ M ในแพคเก็ตที่สอง โดยใช้วิธีการเดียวกันกับที่ดำเนินการมาข้างต้น

```
send_packet([1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
             0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

ด้วยกลยุทธ์ตามที่ระบุไว้ข้างต้น คลีโอพัตราจะคืนค่า: $[1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1]$.

ไอซ่าสามารถเลือกที่จะส่งแพคเก็ตเพิ่มเติมได้ แต่เธอเลือกที่จะไม่ทำดังนั้น

ในขั้นถัดไป เกรดเดอร์เรียกฟังก์ชันต่อไปนี้:

```
receive_message([[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
                  0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0],
                 [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]])
```

บาสม่าจะสร้างข้อความ M กลับมาได้ดังนี้ สำหรับแต่ละแพคเก็ตเธอเลือกบิตแรกที่ปรากฏติดกันสองครั้ง และบิตสุดท้ายที่ปรากฏติดกันสองครั้ง นั่นคือ ในแพคเก็ตแรก เธอจะเลือกบิต $[0,1]$ และจากแพคเก็ตที่สองเธอจะเลือกบิต $[1,0]$ เมื่อนำลำดับทั้งคู่มาต่อกัน เธอจะสามารถสร้างข้อความ $[0,1,1,0]$ ซึ่งเป็นคำตอบที่ถูกต้องของฟังก์ชัน `receive_message` กลับมาได้

สามารถแสดงได้ว่าด้วยกลยุทธ์ของคลีโอพัตราตามที่ระบุมาดังข้างต้น และข้อความที่มีความยาว 4 วิธีการที่ระบุโดยบาสม่าจะสามารถสร้างข้อความตั้งต้น M กลับมาได้ ไม่ว่าค่าของ C จะเป็นอย่างใด อย่างไรก็ตาม วิธีการนี้ไม่ถูกต้องในกรณีทั่วไป

เกรดเดอร์ตัวอย่าง

เกรดเดอร์ตัวอย่างจะไม่ทำงานแบบปรับเปลี่ยนได้ (ไม่ adaptive) นั่นคือ คลีโอพัตราจะใส่ค่าบิตที่เธอควบคุมได้ที่ติดกันด้วยค่า 0 สลับกับ 1 ดังตัวอย่างข้างต้น

รูปแบบข้อมูลนำเข้า: **บรรทัดแรกของข้อมูลนำเข้าจะมีจำนวนเต็ม T ที่ระบุจำนวนสถานการณ์** จากนั้นจะตามด้วยสถานการณ์จำนวน T สถานการณ์ แต่ละสถานการณ์จะระบุในรูปแบบดังนี้:

```
S
M[0]  M[1]  ...  M[S-1]
C[0]  C[1]  ...  C[30]
```

รูปแบบข้อมูลส่งออก: เกรดเดอร์ตัวอย่างจะเขียนผลลัพธ์ของแต่ละสถานการณ์จำนวน T สถานการณ์ในลำดับเดียวกับที่ระบุในข้อมูลนำเข้า ตามรูปแบบนี้:

```
K L
D[0]  D[1]  ...  D[L-1]
```

ในตัวอย่างนี้ K แทนจำนวนของการเรียกฟังก์ชัน `send_packet` D แทนข้อความที่คืนจากฟังก์ชัน `receive_message` และ L แทนความยาว