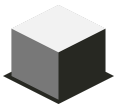
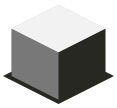
CodeCube the Mini Contest #2

Editorial



Mini Contest 2 by PalmPTSJ

สำหรับข้อนี้เราสามารถไล่เช็คลูกบาศก์ในแต่ละช่อง ตั้งแต่ช่องที่ 1 ถึงช่องที่ N-1 ว่าช่องถัดจากนั้นมีสีที่ซ้ำกันกับช่องนี้หรือไม่ ซึ่งจะใช้เวลาในการประมวลผล O(N)



Modulo Six by bT33

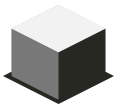
สำหรับโจทย์ข้อนี้ เราให้คุณสร้างจำนวนเต็มบวกที่หารด้วย 6 ลงตัว ที่มีจำนวนหลัก N หลัก และมีผลรวมของแต่ละหลักรวมกันเท่ากับ S

สำหรับจำนวนที่หารด้วย 6 ลงตัว ก็คือจำนวนที่หารด้วย 2 ลงตัว และหารด้วย 3 ลงตัว ซึ่งในกรณีที่เป็นจำนวนที่หารด้วย 2 ลงตัว ก็คือจำนวนคู่ที่มีหลักน่วยเป็น 0, 2, 4, 6 หรือ 8 นั่นเอง ส่วนในกรณีที่เป็นจำนวนที่หารด้วย 3 ลงตัวนั้น จะต้องเป็นจำนวนที่มีผลรวมของแต่ละหลักรวมกันหารด้วย 3 ลงตัว (พิสูจน์ได้โดยใช้ความสัมพันธ์คอนกรูเอนซ์ Congruence Relation)

จากที่กล่าวมาเราสามารถสร้างจำนวนดังกล่าวได้หลายวิธี โดยในที่นี้เราจะขอกล่าวถึง 2 วิธี โดยวิธีแรกนั้น ในขั้นแรกเราจะเลือกหลักสุดท้ายให้เป็นเลขคู่ x โดยที่เลขอีก N - 1 หลักที่เหลือยังสามารถมีผลรวมหลักเท่ากับ S - x ได้หรือก็คือ 1 <= S - x <= 9(N - 1) เมื่อเราได้หลักสุดท้ายที่ต้องการแล้วหลังจากนั้นเราจะเฉลี่ยผลรวม S - x ใส่อีก N - 1 หลักที่เหลือ โดยใช้เวลาเป็น O(N) เช่น

- N = 3 และ S = 18 สมมติเราเลือกหลักสุดท้ายเป็น 2 เหลือผลรวมเป็น 16 นำมาเฉลี่ยวางใน 2 หลักที่เหลือได้ 8 เหลือเศษ 0 นำมาสร้าจำนวนที่ต้องการได้เป็น 882
- N = 5 และ S = 24 สมมติเราเลือกหลักสุดท้ายเป็น 6 เหลือผลรวมเป็น 18 นำมาเฉลี่ยวางใน 4 หลักที่เหลือได้ 4 เหลือเศษ 2 นำมาสร้าจำนวนที่ต้องการได้เป็น 55446

ส่วนอีกวิธีนั้น ให้เราสร้างเลข N หลักที่น้อยที่สุดมาก่อน (1 ตามด้วย 0 จำนวน N-1 ตัว) จากนั้นเราจะไล่เติมเลขในแต่ละหลักจากหลักหน่วยไป โดยหลักหน่วยเราเติมได้ทีละ 2 (เพื่อให้เป็นจำนวนคู่) จนได้ถึง 8 และสำหรับหลักที่เหลือเราสามารถเติมได้ทีละ 1 จนได้ถึงเลข 9 เราก็ไล่เติมไปเรื่อยๆจนกว่าจะครบ S หรือถ้าเติมจนเต็มแล้วยังไม่ครบ ก็แสดงว่าไม่มีคำตอบ วิธีนี้ต้องระวังในกรณีที่ N = 1 ด้วย เนื่องจากหลักหน่วยจะต้องเริ่มจาก 2 แทน เพื่อให้เป็นเลขคู่ วิธีนี้จะใช้เวลา O(S)



Kick Game by bT33

ให้พลังป้องกันของคนที่ i เป็น B[i] และจะเรียกพลังโจมตีของคนที่ i เป็น A[i]

เราสามารถทำ Quick Sum ของพลังป้องกันรวมตั้งแต่คนที่ 1 ถึงคนใด ๆ ก็ตาม ในเวลา O(N) โดยต่อไปเราจะเรียกผลรวมของพลังป้องกันรวมตั้งแต่คนที่ 1 ถึงคนที่ i ว่า qs[i]

สำหรับแต่ละคน เราจะทำการ binary search ค่า R เพื่อหาค่า R ที่มากที่สุด ที่พลังป้องกันรวมของคนทางซ้ายไป R คนและขวาไป R คนมีค่าไม่เกินพลังโจมตีของตน โดยช่วงของการทำ binary search ก็คือ [0, (n-1)/2] เราจะทำ binary search ทั้งหมดใน O(log N) โดยแต่ละครั้งใช้เวลาการหาพลังป้องกันรวมทั้ง 2R คนใดๆ เป็น O(1) เพราะเราทำ Quick Sum ไว้แล้ว (เราสามารถหาพลังป้องกันของคนตั้งแต่ช่วง [i, j] ใดๆได้ โดยการหาพลังป้องกันของคนในช่วง [1, j] ลบ พลังป้องกันของคนในช่วง [1, i-1] หรือพูดง่ายๆก็คือ พลังป้องกันของคนตั้งแต่ช่วง [i, j] มีค่าเท่ากับ qs[j] - qs[i-1]) ดังนั้น คน 2R คนรอบๆคนที่ i ก็คือ qs[i+R] - qs[i-R-1] - B[i]

แต่เนื่องจากคนของเรานั่งเป็นวงกลม ดังนั้น เราอาจจะต้องเช็คว่าถ้าเลยไปทางใดทางหนึ่ง ให้ทำการวนกลับมานับที่อีกด้านเพิ่ม แต่มีวิธีอื่นที่ง่ายกว่า ก็คือในตอนแรก ให้เราคิดว่ามีคนทั้งหมด 3n คน โดยเป็นทั้ง n คนนั้นต่อกันสามรอบ แล้วทำการ binary search ทั้ง n คนนั้นจากตรงกลาง เช่น ถ้าเรามีคนทั้งหมด 5 คน ก็ให้สร้างคน 15 คนตามรูปข้างล่างนี้ และทำการ binary search จากคนในกรอบสีดำ 5 คนนั้นเท่านั้น ซึ่งจะทำให้การหาพลังป้องกันรวมที่วนรอบวงกลมง่ายกว่ามาก ก็คือ qs[n + i + R] - qs[n + i - R - 1] - B[i]

1 2 3 4 5 [1 2 3 4 5] 1 2 3 4 5

โดยสรุปแล้ว เราจะใช้เวลาประมวลผลทั้งหมด O(N log N) และใช้หน่วยความจำ O(N)