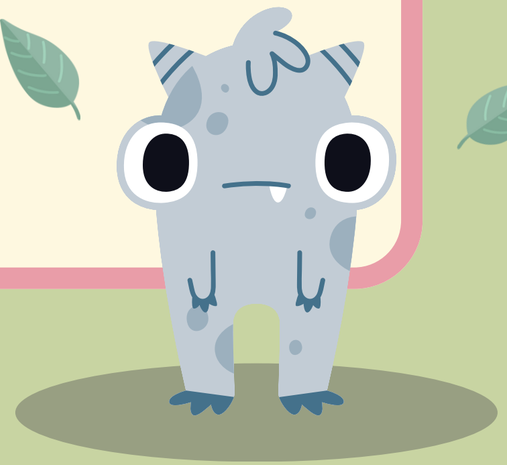
Convex Hull

Andrew Chain

Data structure ในการเก็บข้อมูลเป็นแบบใด

1) อาร์เรย์ (Array)

- ใช้เก็บจุดทั้งหมดในระนาบ 2 มิติ
- จุดจะถูก จัดเรียง ตามค่าแกน x (และแกน y หากค่า x เท่ากัน)
  เรียงลำดับจุดจากซ้ายไปขวา (ค่า x น้อยไปหาค่า x มาก)
  หากมีจุดที่ค่า x เท่ากัน:
  ให้เรียงตามค่า y จากน้อยไปมาก (ล่างไปบน)



Data structure ในการเก็บข้อมูลเป็นแบบใด

2) สแต็ก (Stack)

- ใช้เก็บจุดที่เป็นส่วนหนึ่งของ Convex Hull ชั่วคราว
- จุดสามารถเพิ่ม (Push) หรือเอาออก (Pop) ได้ตามการตรวจสอบว่าเป็นส่วนหนึ่งของ Convex Hull

Algorithm มีเป้าหมายอะไร

สร้าง Convex Hull

- สร้าง เส้นรอบรูปนูนที่สุด ที่ครอบคลุมจุดทั้งหมดในระนาบ 2 มิติ
- ใช้การจัดเรียงจุดและแบ่งเป็น Lower Hull และ Upper Hull เพื่อสร้างเส้นรอบรูป
- ทำงานอย่างมีประสิทธิภาพในเวลา O(N log N)
- ใช้ในปัญหาที่ต้องการเส้นรอบนอก เช่น คำนวณพื้นที่ หรือวิเคราะห์จุดรอบนอกสุด

การทำงานขั้นตอนของ Algorithm

1) จัดเรียงจุด

- ก่อนเริ่ม: ให้จัดจุดทั้งหมดเรียงตาม ค่าแกน x จากซ้ายไปขวา (ถ้าค่า x เท่ากันให้เรียงตามค่า y)

เช่น:

ถ้ามีจุด [(2, 3), (1, 1), (3, 4), (0, 0)]
เมื่อจัดเรียงจะเป็น: [(0, 0), (1, 1), (2, 3), (3, 4)]

2) สร้าง "Lower Hull"

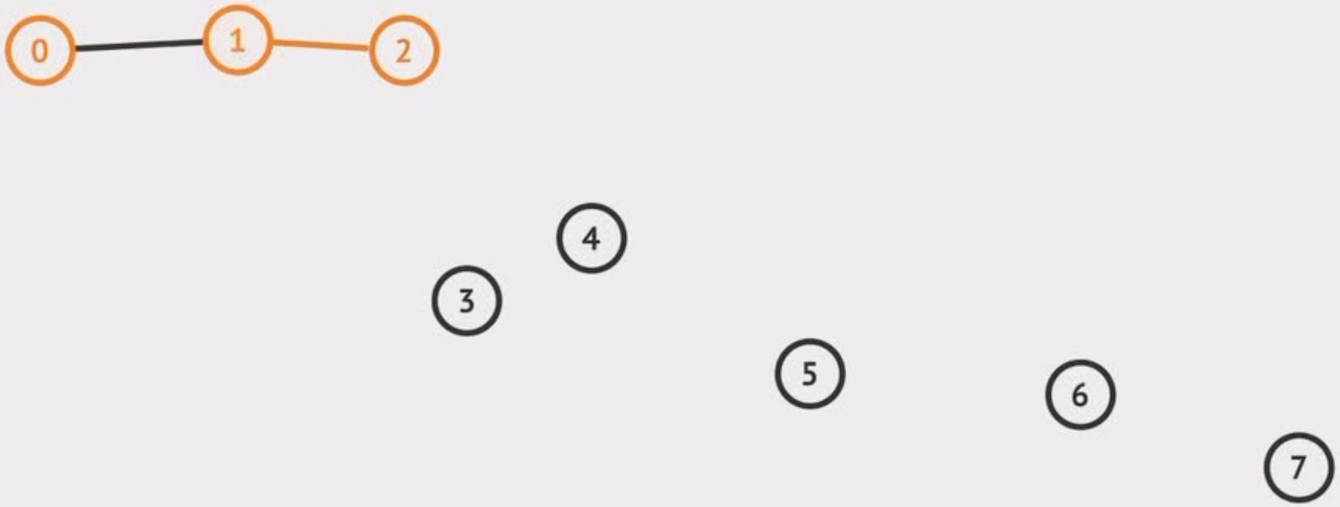
- เริ่มจากจุดซ้ายสุด (เล็กที่สุด) ไปยังจุดขวาสุด (ใหญ่ที่สุด)
- เพิ่มจุดทีละจุดลงในเส้นรอบรูป (เริ่มต้นด้วย 2 จุดแรก)
- ตรวจสอบว่าเส้นที่เกิดจากจุดสามจุดล่าสุด หัน "ออกด้านนอก" หรือ "เข้าด้านใน"
- ใช้ฟังก์ชัน ccw (Counter Clockwise)
  - ถ้าหัน เข้าด้านใน → ให้ลบจุดก่อนหน้าออก เพราะมันไม่ใช่ส่วนหนึ่งของ Convex Hull
  - ถ้าหัน ออกด้านนอก → เก็บจุดนั้นไว้ใน Convex Hull
- ทำซ้ำจนถึงจุดขวาสุด → จะได้ Lower Hull

3) สร้าง "Upper Hull"

- เริ่มจากจุดขวาสุด (ใหญ่ที่สุด) ย้อนกลับไปยังจุดซ้ายสุด (เล็กที่สุด)
- ใช้หลักการเดียวกับตอนสร้าง Lower Hull
- ทำซ้ำจนถึงจุดซ้ายสุด → จะได้ Upper Hull

4) รวมผลลัพธ์

- นำจุดจาก Lower Hull และ Upper Hull มารวมกัน
- ผลลัพธ์จะเป็น Convex Hull ครบถ้วน


0
1
2
3
4
5
6
7

Convex Hull

Andrew Chain