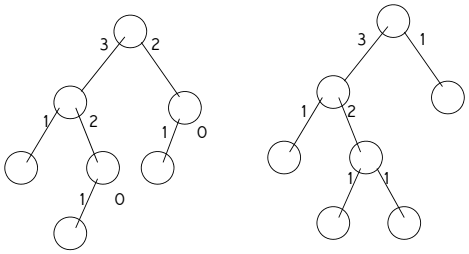
AVL Tree

นิยามของ AVL Tree

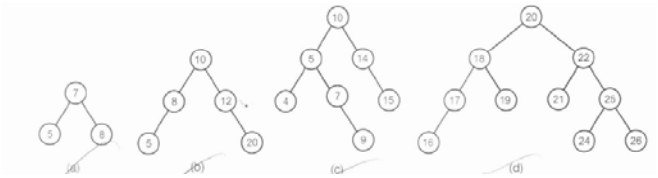
- ทรีใดๆ ก็ตามจะจัดว่าเป็น AVL Tree ก็ต่อเมื่อทรีนั้นมีคุณสมบัติต่อไปนี้ครบทั้ง 3 ข้อ
  - ทรีนั้นต้องเป็นไบนารีทรี
  - ทรีนั้นต้องเป็นไบนารีเซร์ชทรี
  - ทรีนั้นต้องเป็น Blance Tree
- สำหรับคุณสมบัติสองข้อแรกสามารถตรวจสอบได้ง่าย ส่วนคุณสมบัติข้อที่สามเป็นคุณสมบัติในเรื่องความสมดุลของทรี หมายถึง ทุกโหนดในทรีจะต้องมีความสูงของ Left subtree และ Right subtree ต่างกันไม่เกิน 1 หรือ "AVL Tree คือ ไบนารีเซร์ชทรีที่ทุกๆ โหนดในทรีจะต้องมีค่าความสูงของ Left subtree และ Right subtree ต่างกันไม่เกิน 1

การตรวจสอบความสมดุลของ AVL Tree

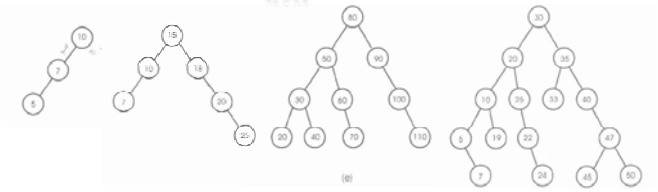
1. พิจารณาโหนดของทรีที่จะตรวจสอบทีละโหนดจากล่างขึ้นบน ทีละ Level ไปตามลำดับ
2. ในแต่ละโหนดจะมีค่า H_L และ H_R อย่างละเท่าไร
3. ถ้า $|H_L-H_R|$ ได้ค่ามากกว่า 1
   3.1 ใช่ แสดงว่าไม่เป็น AVL Tree (ใช้วิธีการปรับให้เป็น AVL Tree)
   3.2 ไม่ใช่ ให้ขยับไปพิจารณาโหนดถัดไป วนย้อนไปพิจารณาข้อ 2
4. วนทำเช่นนี้ไปจนกระทั่งตรวจสอบครบทุกโหนด ถ้าค่า $|H_L-H_R|$ น้อยกว่าหรือเท่ากับ 1 ทุกโหนด สามารถสรุปได้ว่าเป็น AVL Tree



AVL Non-AVL



AVL



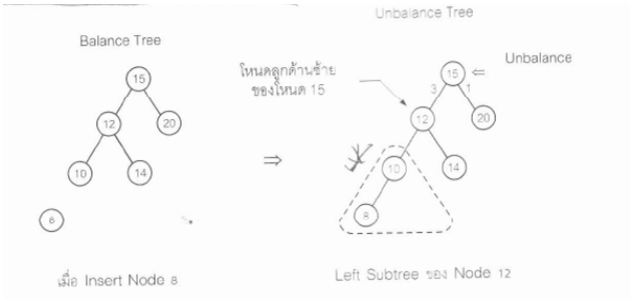
NON-AVL

ทรีที่ไม่สมดุล(Unblance Tree)

- เกิดจากการแทรกโหนด หรือลบโหนด จากทรีที่สมดุลอาจกลายเป็นทรีไม่สมดุลได้ ซึ่งรูปร่างของทรีที่ไม่สมดุลเกิดขึ้นได้ 4 รูปแบบ คือ
  - Left of Left
  - Right of Right
  - Right of Left
  - Left of Right

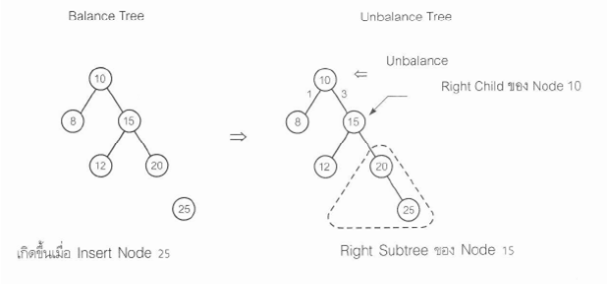
ทรีที่ไม่สมดุล

- Left of Left: เกิดจากการแทรกโหนดเข้าไปทางด้านซับทรีฝั่งซ้ายของโหนดลูกด้านซ้าย



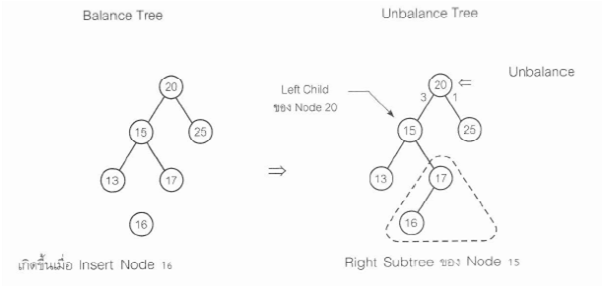
ทรีที่ไม่สมดุล

- Right of Right : เกิดจากการแทรกโหนดเข้าไปทางด้านซับทรีฝั่งขวาของโหนดลูกด้านขวา



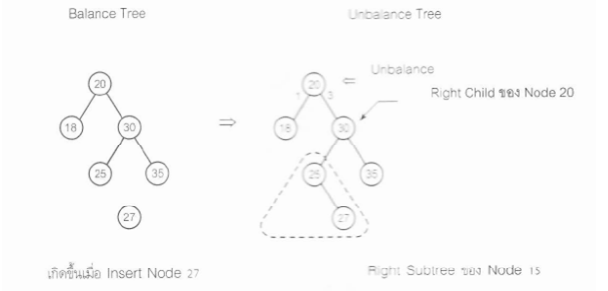
ทรีที่ไม่สมดุล

- Right of Left: เกิดจากการแทรกโหนดเข้าไปทางด้านซับทรีฝั่งขวาของโหนดลูกด้านซ้าย



ทรีที่ไม่สมดุล

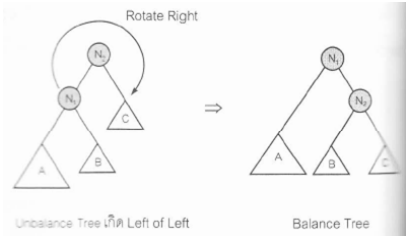
- Left of Right : เกิดจากการแทรกโหนดเข้าไปทางด้านซับทรีฝั่งซ้ายของโหนดลูกด้านขวา



การทำไบนารีเสิร์ชทรีให้กลายเป็น AVL Tree

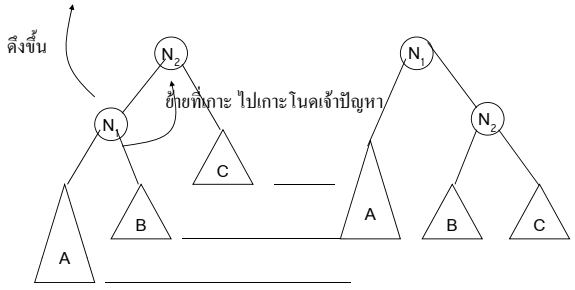
- การแปลงมีอยู่ 2 แบบ การเลือกใช้แบบใดนั้น ขึ้นอยู่กับรูปแบบความไม่สมดุลของทรี
  - แบบหมุนครั้งเดียว(Single Rotation): ในกรณี Left of Left และ Right of Right เท่านั้น
  - แบบหมุนสองครั้ง(Double Rotation): ในกรณี Right of Left และ Left of Right เท่านั้น

Single Rotation: Left of Left



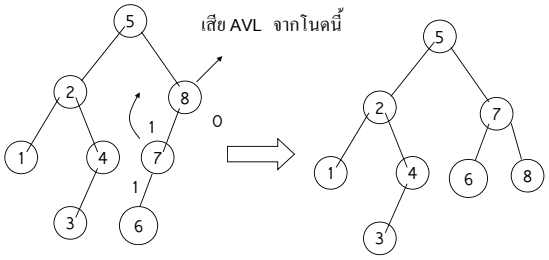
- จากภาพจะเห็นว่า Left Subtree A มีขนาดใหญ่เป็นสาเหตุให้ทรีเอียงหนักมาทางซ้าย จึงต้องแก้ปัญหาโดยการหมุนขวา 1 ครั้ง ผลลัพธ์ที่ได้คือ จะได้ทรีที่มีความสมดุลเกิดขึ้นมา ในรูปแสดงว่าโหนด N1 มีค่าน้อยกว่า N2 (โหนดที่เป็นรูทจะเปลี่ยนไป แต่ส่วนอื่นที่ไม่มีการหมุนยังเหมือนเดิม

Single rotation หลังจากการ insert

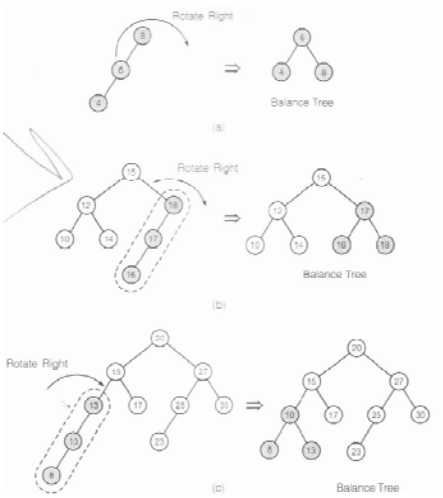


แก้ได้แล้วก็ไม่ต้อง rotate ที่ไหนต่ออีก

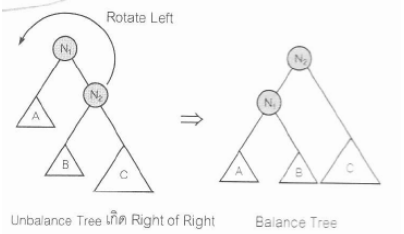
ตัวอย่าง 1



ตัวอย่างการทำ single rotation :Left of Left

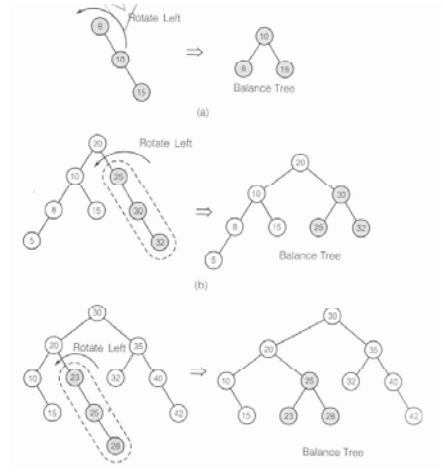


Single Rotation: Right of Right

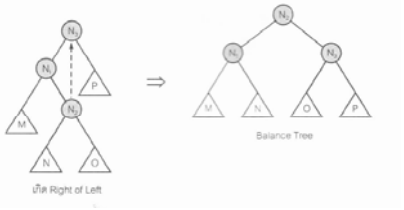


- จากภาพจะเห็นว่า Right Subtree C มีขนาดใหญ่ เป็นสาเหตุให้ทรีเอียงหนักมาทางขวา จึงต้องแก้ปัญหาโดยการหมุนขวา 1 ครั้ง ผลลัพธ์คือ จะได้ทรีที่มีความสมดุล ในรูปแสดงว่าโหนด N1 มีค่าน้อยกว่า N2 (โหนดที่เป็นรูทจะเปลี่ยนไป แต่ส่วนอื่นที่ไม่มีการหมุนยังเหมือนเดิม

ตัวอย่างการทำ single rotation :Right of Right

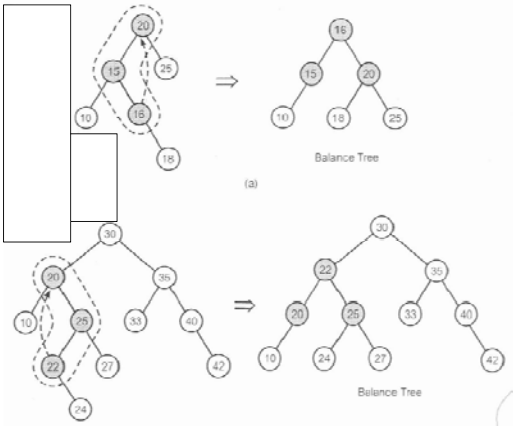


การทำ Double Rotation: Right of Left



- จากภาพ Subtree มีขนาดเท่ากัน แต่ทรีไม่สมดุลจะเกิดอยู่บริเวณโหนดล่วนที่อยู่ด้านบน แก้ปัญหาโดยการหมุนทรี เมื่อหมุนแล้ว N2 จะถูกดึงขึ้นมาเป็นรูทแทนโดย Subtree ที่เหลือก็จะต้องเปลี่ยนไป เชื่อมต่อกันโหนดที่จะให้คงคุณสมบัติของไบนารีเสิร์ชทรีเอาไว้

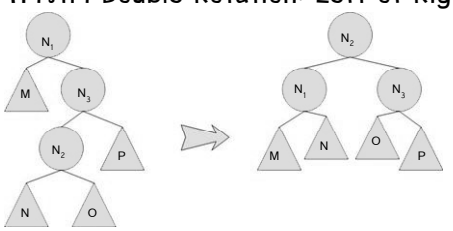
ตัวอย่างการทำ Double rotation :Right of Left



การเลือกโหนดที่จะหมุน

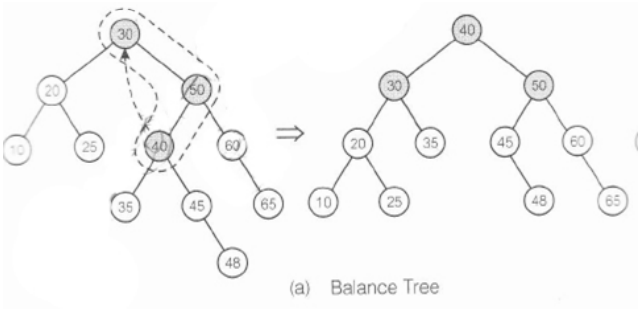
- ดูว่าเกิดปัญหาความไม่สมดุลที่โหนดไหน เอาโหนดนั้นมาเป็นโหนดเริ่มต้นของกลุ่ม
- จากโหนดเริ่มต้นนับลงไปอีก 2 โหนด จะได้กลุ่มโหนด 3 โหนด โดยทิศทางในการนับลงมา คือ นับลงมาทางด้าน Child ที่ทำให้เกิดปัญหาความไม่สมดุล

การทำ Double Rotation: Left of Right



- จากภาพ Subtree มีขนาดเท่ากัน แต่ทรีไม่สมดุลจะเกิดอยู่บริเวณโหนดส่วนที่อยู่ด้านบน แก้ปัญหาโดยการหมุนทรี เมื่อหมุนแล้ว N2 จะถูกดึงขึ้นมาเป็นรูทแทนโดย Subtree ที่เหลือก็จะต้องเปลี่ยนไปเชื่อมต่อกันโหนดที่จะให้คงคุณสมบัติของไบนารีเสิร์ชทรีเอาไว้

ตัวอย่างการทำ Double rotation : Left of Right



การสร้าง AVL Tree จากโหนด

1. ให้โหนดแรกเป็นรูทของ AVL Tree
2. แทรกโหนดตัวถัดไปเข้าไปใน AVL Tree
3. ตรวจสอบว่ายังคงคุณสมบัติของ AVL Tree อยู่หรือไม่

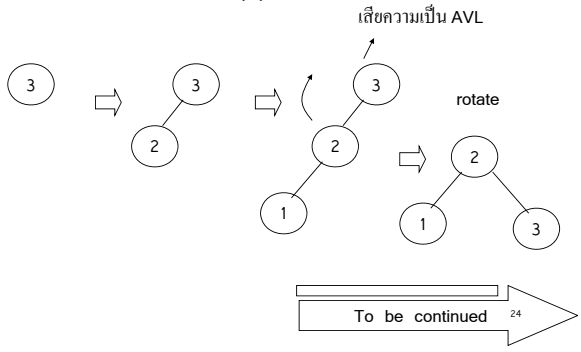
   3.1 ถ้าใช่ ให้แทรกโหนดถัดไปเข้ามาอีกแล้วไปขั้นตอน 3

   3.2 ถ้าไม่ใช่ ให้ใช้เทคนิค Single หรือ Double Rotation แก้ให้กลับมาเป็น AVL Tree จากนั้นแทรกโหนดถัดไป แล้วไปขั้นตอนที่ 3
4. ให้ทำเช่นนี้ไปจนสามารถแทรกโหนดเข้าไปใน AVL Tree ได้จนครบทุกโหนด

ตัวอย่าง 2 สร้าง AVL Tree จากโหนดต่อไปนี้

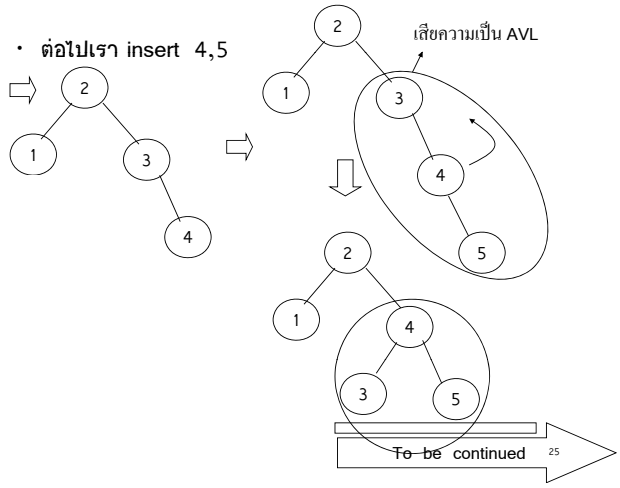
3, 2, 1, 4, 5, 6, 7

- เริ่มจากไม่มีอะไร เรา insert 3,2,1

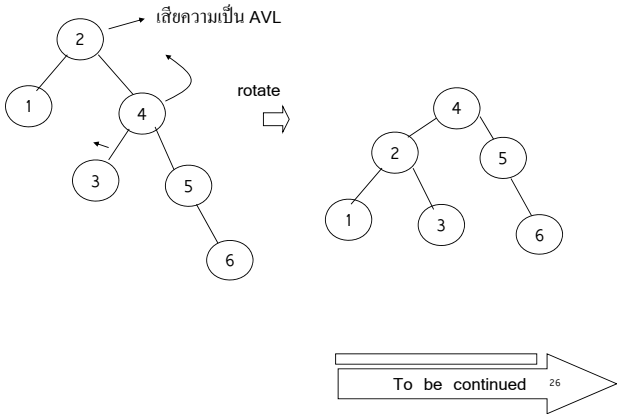


To be continued

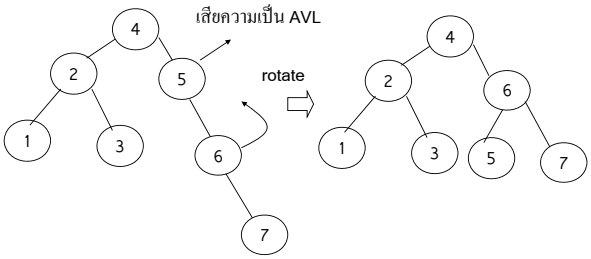
- ต่อไปเรา insert 4,5



- ต่อไปเรา insert 6

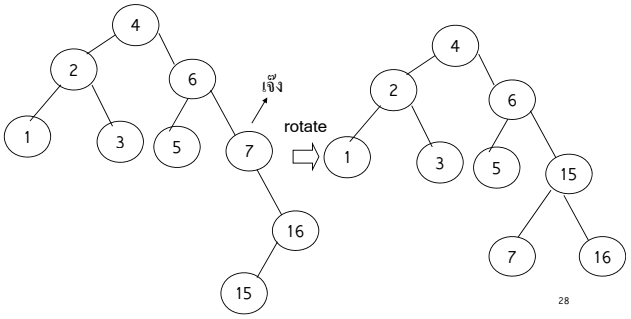


- ต่อไปเรา insert 7

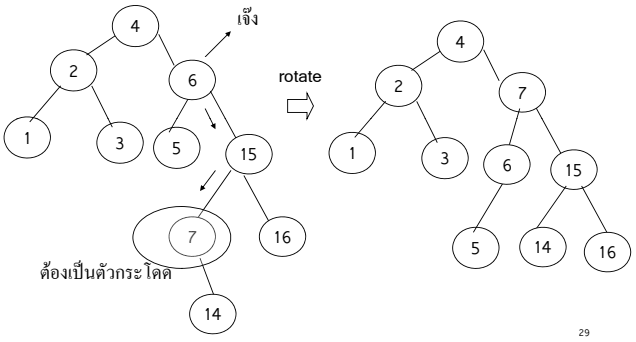


ตัวอย่างเพิ่มเติม

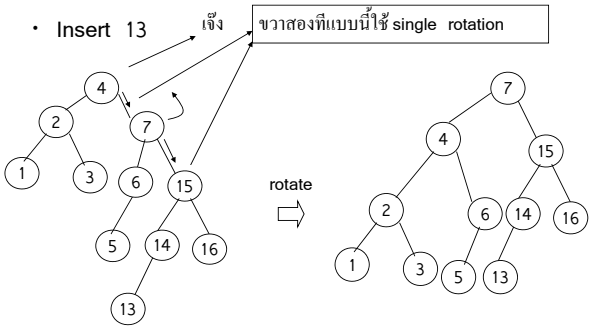
- Insert 16,15 เสียความเป็น AVL ตอน insert 15



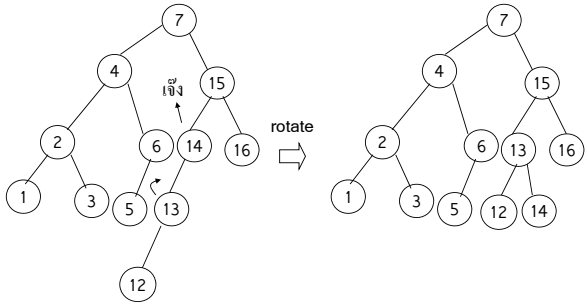
- Insert 14



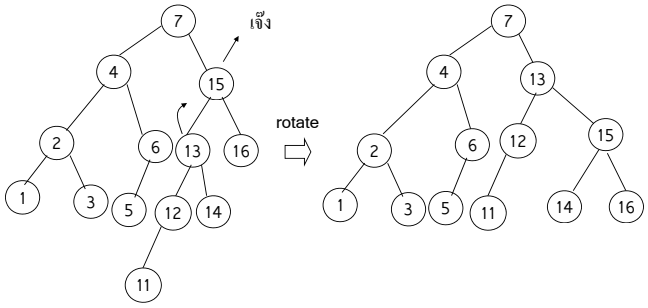
- Insert 13



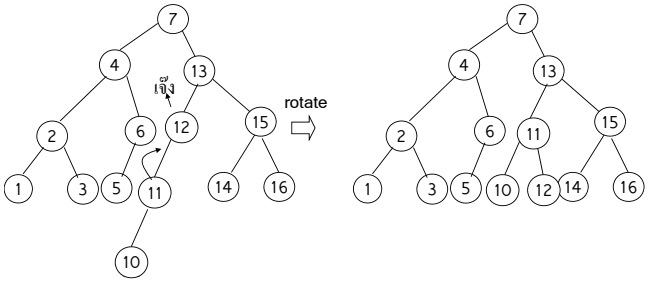
- Insert 12 ก็ทำให้ต้อง single rotate อีก



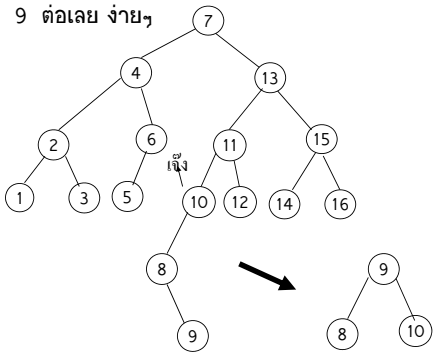
- Insert 11 ก็ทำให้ต้อง single rotate อีก



- Insert 10 ก็ทำให้ต้อง single rotate อีก



- Insert 8 ไม่มีอะไรเกิดขึ้น
- Insert 9 ต่อเลย ง่ายๆ



- ให้แสดงวิธีการสร้าง AVL Tree จากโหนดต่อไปนี้

รหัสนักศึกษา+0+วัน+เดือน+ปีเกิด

เช่น รหัส 51152792055+0+10+07+25+30 จะได้

51,15,27,92,05,50,10,07,25,30

- ระบุ parent node,child node, sibling node,leaf node
- หาผลลัพธ์จากการเยือนโหนดต่างๆ ด้วยวิธี Preorder, Inorder และ Postorder