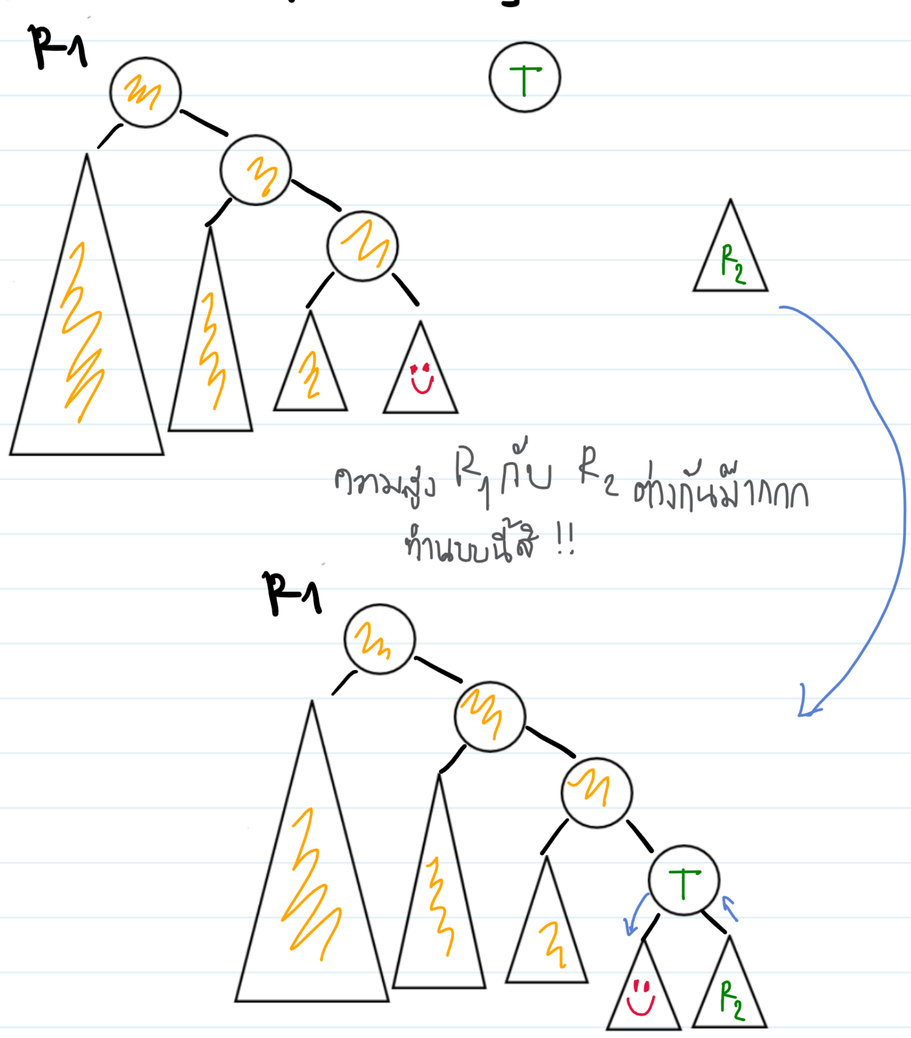
Data 17/9/2021

Merge AVL tree ที่ 2 ต้นมีความสูงต่างกันมากกว่า 1 ทำไง??

→ Mergeกับ RST ของ R_1 จนกว่าจะ Merge กันได้



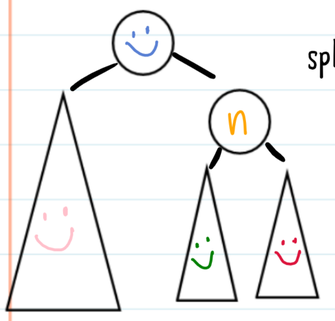
AVLTreeMergeWithRoot (R_1, R_2, T)

```
if |R1.height − R2.height| ≤ 1:
    MergeWithRoot (R1, R2, T)
    T.height ⟵ max(R1.height, R2.height)+1
    return T
else if R1.height > R2.height
    R' ⟵ AVLTreeMergeWithRoot (R1.right, R2, T)
    R1.right ⟵ R'
    R'.parent ⟵ R1
    Rebalance (R1)
    return root
```

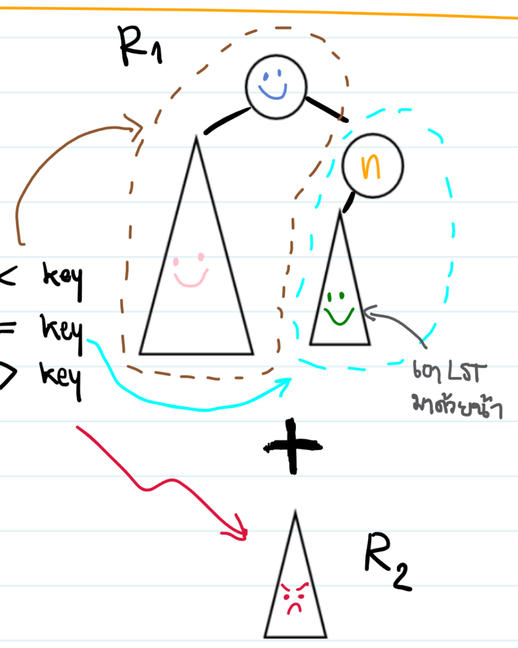
else if R_1.height < R_2.height
↳ Merge ด้านซ้ายแล้วจะบอกให้ (HW) :=

สรุป → $O(|R_1.height - R_2.height| + 1)$
ความต่างมากที่สุด $O(\log n)$

Split (BST)



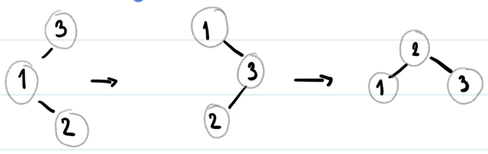
split key(n)
ถ้า node ที่อ้างถึงมา < key
" " = key
" " > key



Implimentation

```
if R is null
    return (null, null)
else if x < R.key :
    (R1, R2) ⟵ split (R.left, x)
    R3 ⟵ MergeWithRoot (R2, R.right, R)
    return (R1, R3)
else if x ≥ R.key :
    (R1, R2) ⟵ split (R.right, x)
    R4 ⟵ MergeWithRoot (R.left, R1, R)
    return (R4, R2)
```

* ถ้าอยากให้เป็น AVL tree
→ เปลี่ยน MergeWithRoot เป็น AVLMergeWithRoot
→ Rebalance ด้วยนะ
→ O(log n)



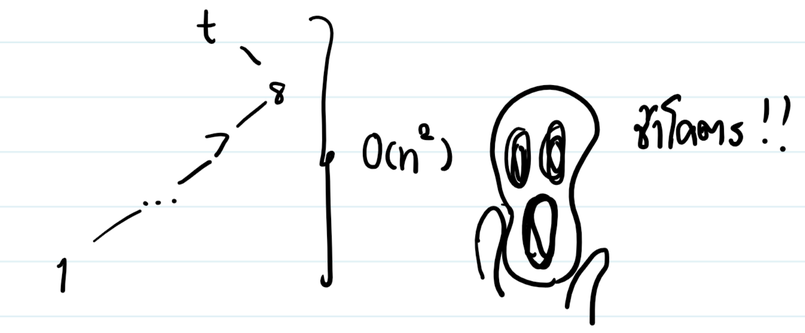
Splay tree

นิยาม : BST ไม่ต้อง rebalance ตลอดเวลา ข้อมูลที่เพิ่งเข้าถึงจะถูกพามาที่ root

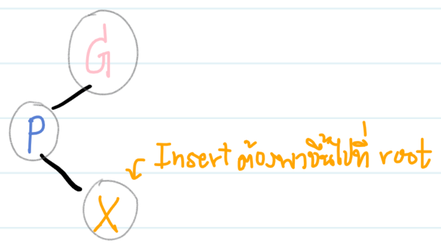
→ นำ node ที่กำลังค้นหามาไว้ที่ root
→ ถ้า node นั้นโดนลบไปแล้วให้พาแม่มันมาที่ root

Insert at root

แบบตกกะปิ

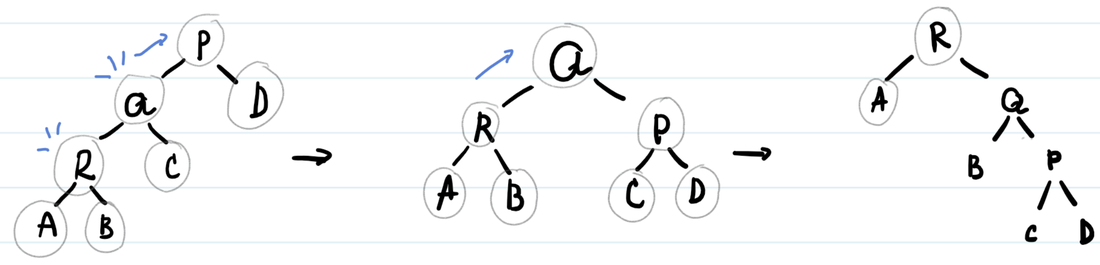


ทำไงดี ?
→ Insert เลยแล้ว splay node นั้นขึ้นมาที่ root



X เป็นลูก → ขึ้นไป zig เดียว
X เป็นหลาน++ → zig zig or zigzag

zigzig : zig(R.parent) ⟶ zig(R)



* ให้ใช้ while loop เพราะถ้าใช้ recursive มีโอกาส StackOverflow

zigzag : zig(R) ⟶ zig(R)

Delete

→ หา node ที่จะลบมาไว้ที่ root
→ ลบเลย (เกิด Sub tree 2 ต้น)
→ splay max ของ LST มาที่ root
→ อีกทาง : Splay min ของ RST มาที่ root (ไม่จำเป็น)
→ เชื่อม node โลด