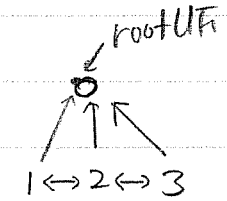
1

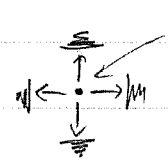
Create

- rank = 0
- degree = 0

create uf → tree
→ uf

✓ **Make set**

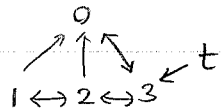


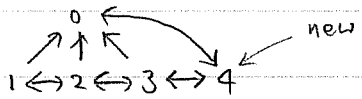


degree = 0
rank = 1
item → 4

Case 2

t ≠ nil

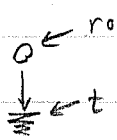




rootUF → degree += 1

Case 1

t = nil



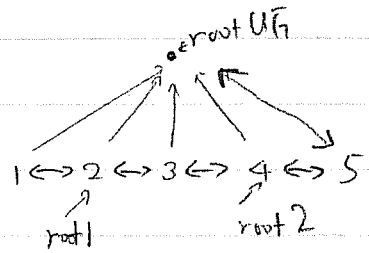
rootUF → degree = 1

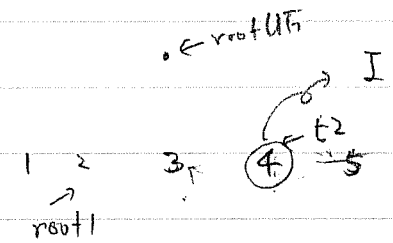
2.

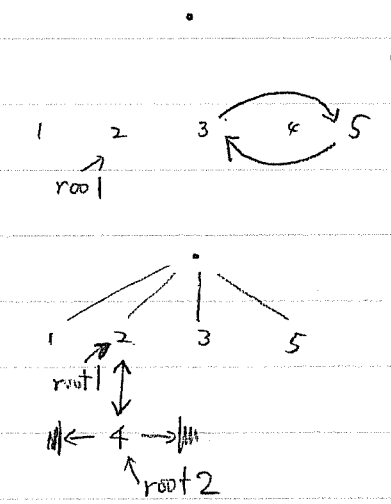
1. Let root1 → rank ≥ root2 → rank.

Cases root1, root2 ranks.

Case 1 vs. 1.

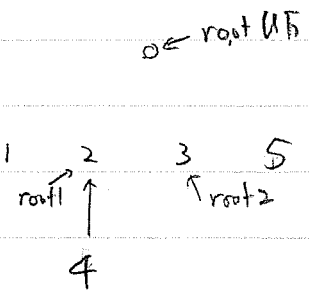


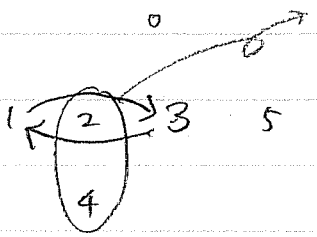


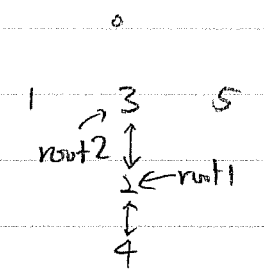


3

Case 2 vs. 1





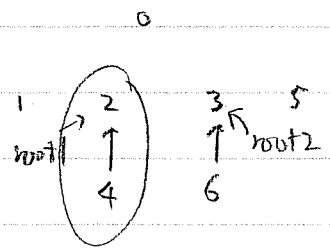


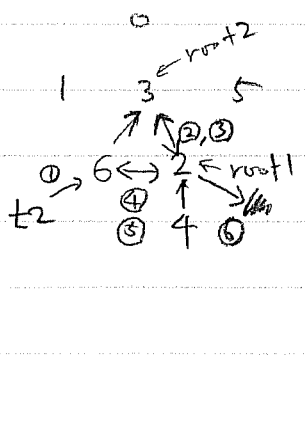
rootUF → degree (−1)

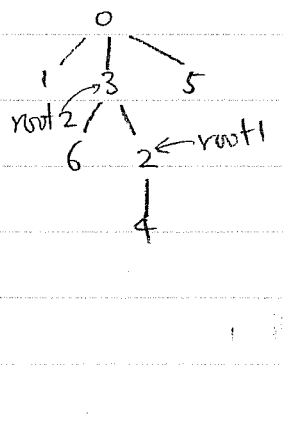
roo2 → rank = 3

→ degree = 1

Case 2 vs. 2.







rootUF → degree (−

root2 → rank = 3

→ degree (+

root 2 and ro

could have

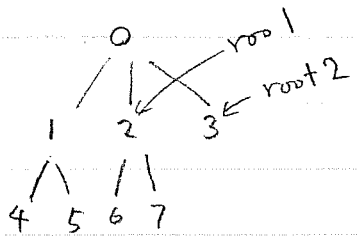
multiple chil

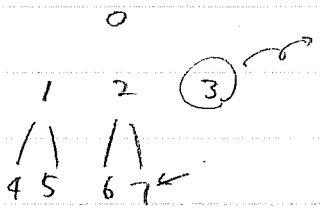
or

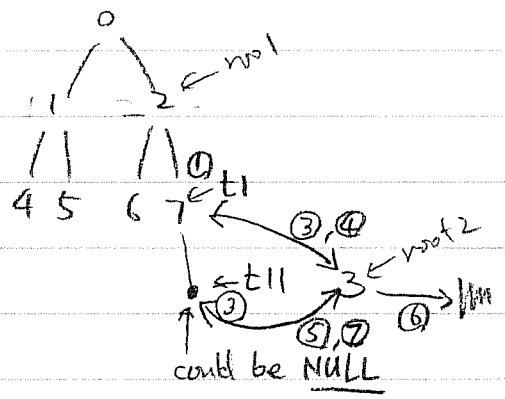
degree > [

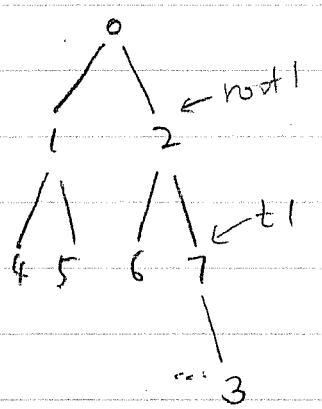
4. Case 3-1, 4-1, 5-1, ..., n-1

($n \geq 3$)







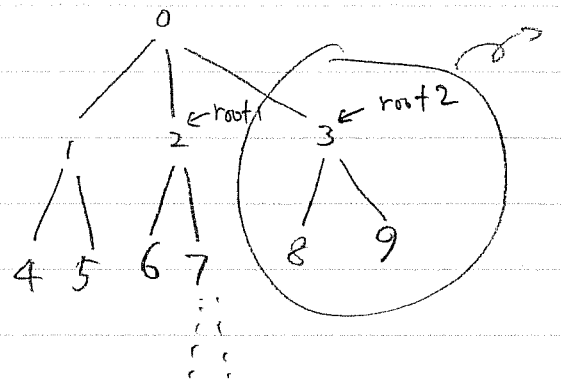


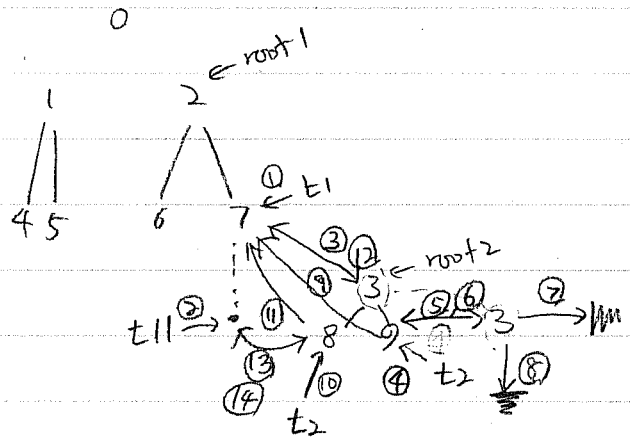
rootUn → degree (-1)

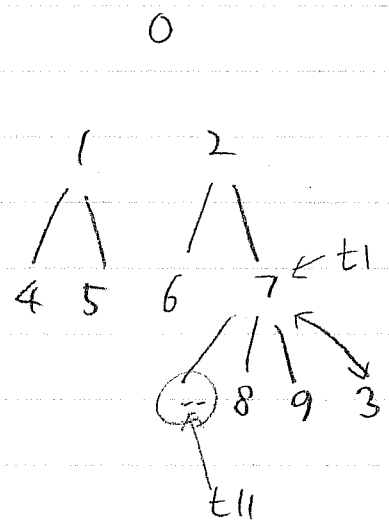
t1 → rank ≥ 2

→ degree (+1)

5. Case 3-2, 4-2, ...







root(UF) → degree (-1)

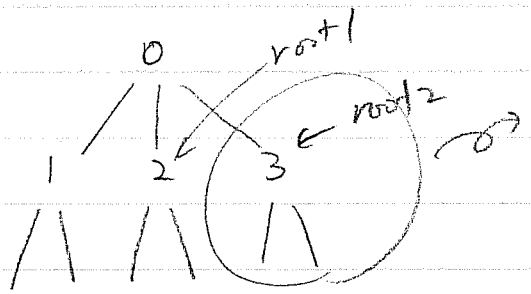
t1 → degree (+ root2→degree)
+1

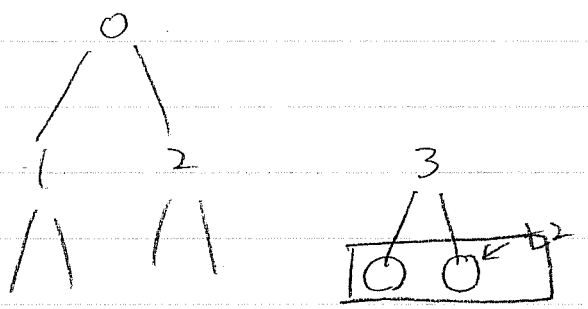
t1 → rank ≥ 2

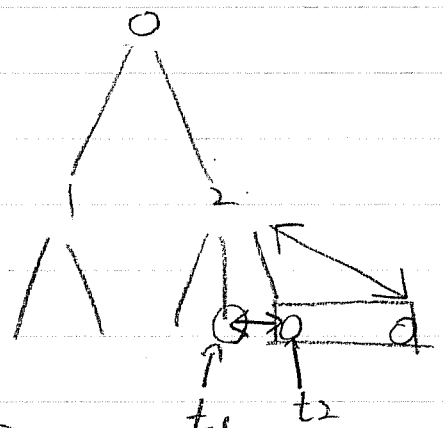
if t11 = nil, t1→rank = 2

t11 ≠ nil, t1→rank sta

6. Case 3-3, 4-4, ...

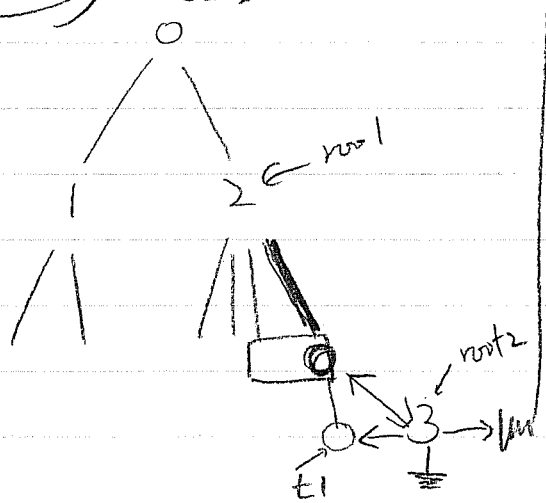






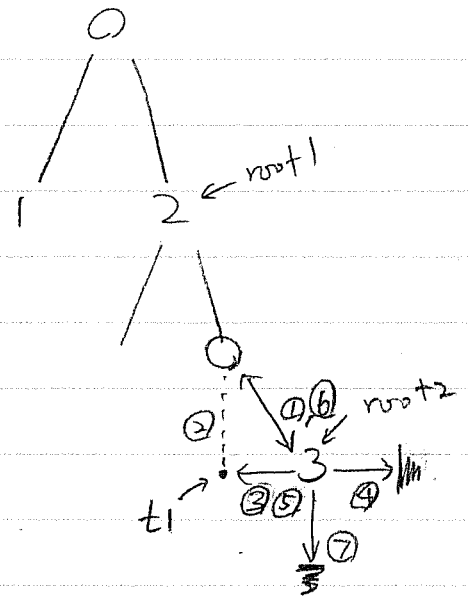
Wider because it is tall

Taller because it is



7.

wider



root2 → rank = 1

→ degree = 0

root1 → down → degree (+

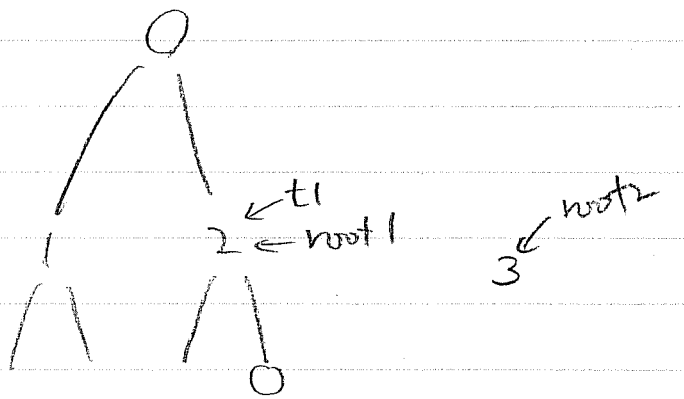
→ rank

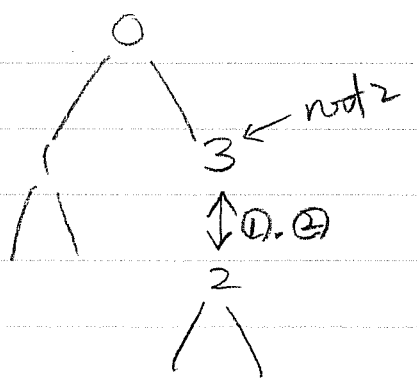
↳ 2 if t1

Stays put if t

root1 → degree (+ root2 → degre

taller





root2 → rank = root → rank

root2 → degree = 1

root1 → degree (+ root2 → deg