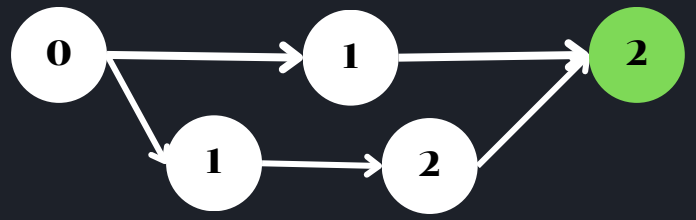
DAY 6

ADVANCE PROBLEM 2

LeViAtHaN

เฉลย GAME ROUTES



เป็น topological sort ผสมกับ DP
การหาจำนวนทางที่มาได้ทั้งหมดของ node u มาจากการคูณทางที่มาได้ของ node ที่ชี้เข้า u ทั้งหมด
แต่เราจะแน่ใจได้ไงว่าเมื่อ DFS ไปถึง u ทางที่ชี้มาจะเป็นระยะที่ถูกแล้ว

เราเลยใช้ topo มา track เพื่อมั่นใจว่าเราจะไปยัง node ที่พร้อมแล้วเท่านั้น

แล้วพอได้แล้วก็คูณ node ที่ไปถึงได้จาก reversed path ทั้งหมดนั้นเอง

เฉลย ROUND TRIP 2

ไม่ยากอะไร จะใช้ Kosaraju หรือ trajan ก็ได้ หรือแค่ DFS โต้งๆแต่เก็บว่ามีตัวไหนอยู่ใน stack ก็ได้
ตอน DFS เราเก็บ on stack ไว้และเก็บ pa ไว้ จะได้เช็คว่าผ่านมากี่เมืองแล้ว และ เอาไว้ย้อน node ได้

DP ON TREE

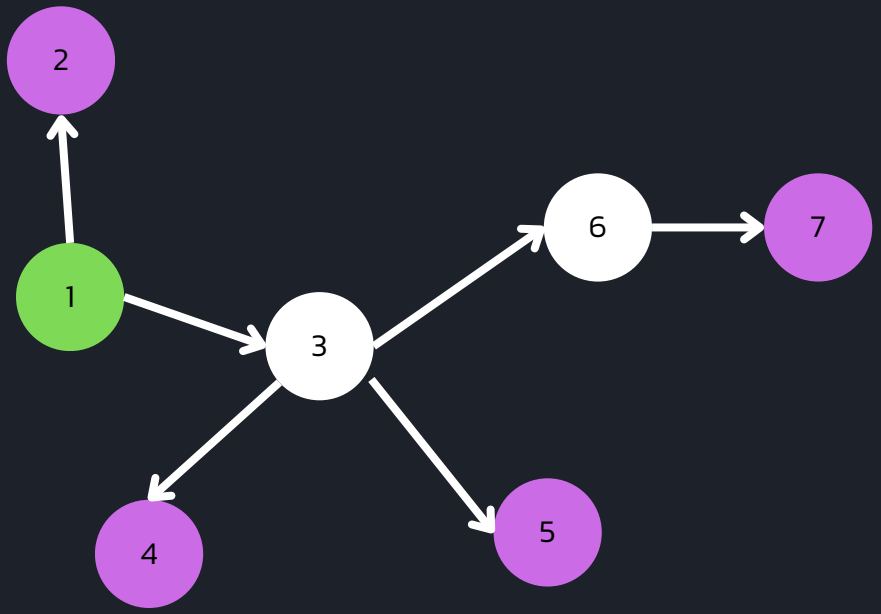
LeViAtHaN

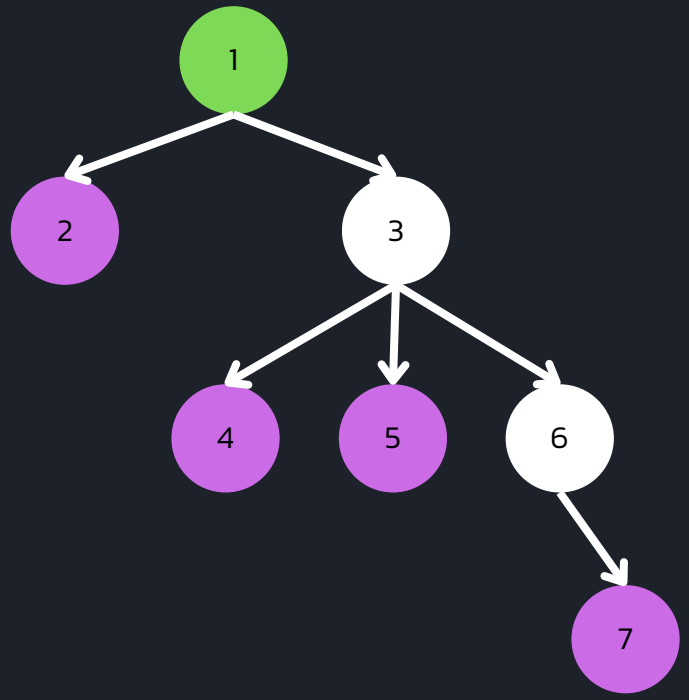
DP ON TREE คืออะไร

Tree เป็น กราฟประเภทหนึ่ง ที่มี N Node และ N-1 edge, ไม่มี cycle, เป็น connected graph

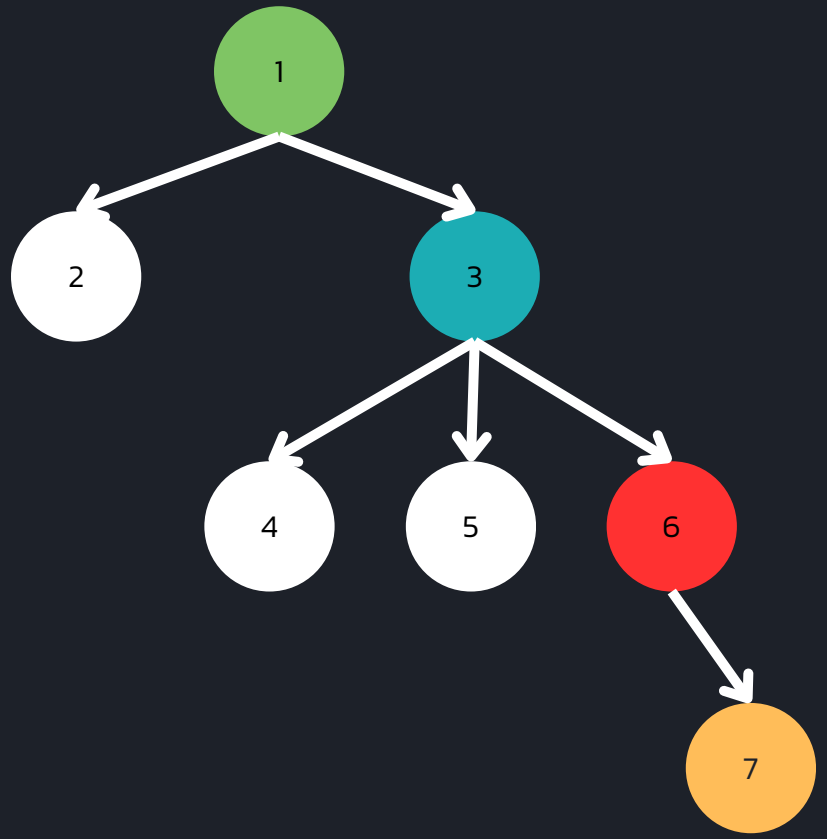
root node ที่อยู่บนสุดของ tree, ไม่มี parent

leaf node ที่มี degree แค่ 1 / อาจมองว่าเป็น node ที่ไม่มี child ก็ได้





DP ON TREE คืออะไร



Parent upper neighbors, ทำ graph travelsal มาถึงก่อน
ถ้าเป็นกราฟมีทิศทาง ก็คือ node ที่ชี้เข้า

Child lower neighbors, ทำ graph travelsal มาถึงหลัง
ถ้าเป็นกราฟมีทิศทาง ก็คือ node ที่ชี้ออก

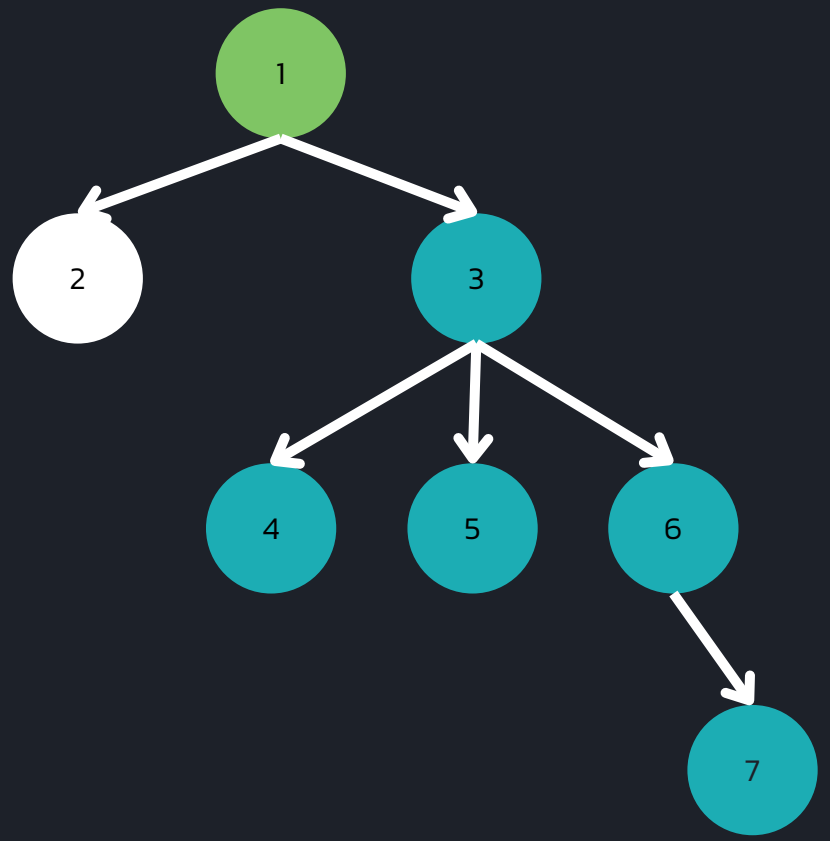
ancestors parent ของ parent อีกที

```
int sol(int u){
    int temp=0;
    for(auto v:path[u]){
        temp+=sol(v);
    }
    child[u]=temp;
    return temp + 1;
}
```

นับจำนวน child ของ node ใดๆ

LeViAtHaN

DP ON TREE คืออะไร



subtree หน่วยย่อยๆของ Tree ที่เป็น Tree เล็กๆ

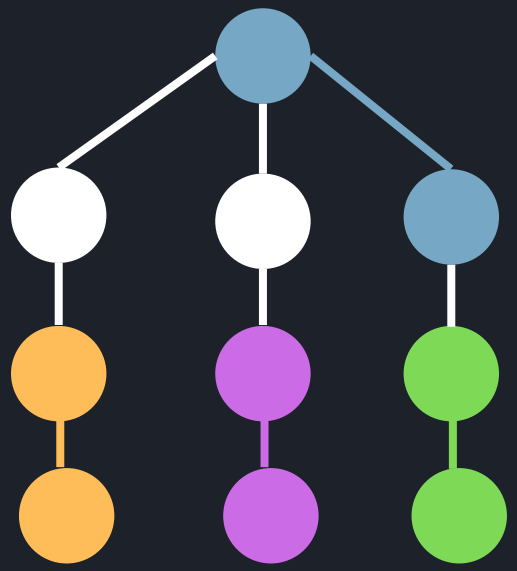
Depth ระยะห่างของของ node จาก root

CSES Tree Matching

[20 min]

LeViAtHaN

CSES Tree Matching



จะจับคู่ node ที่มี path เชื่อมกันยังไงให้มีคู่มากสุด

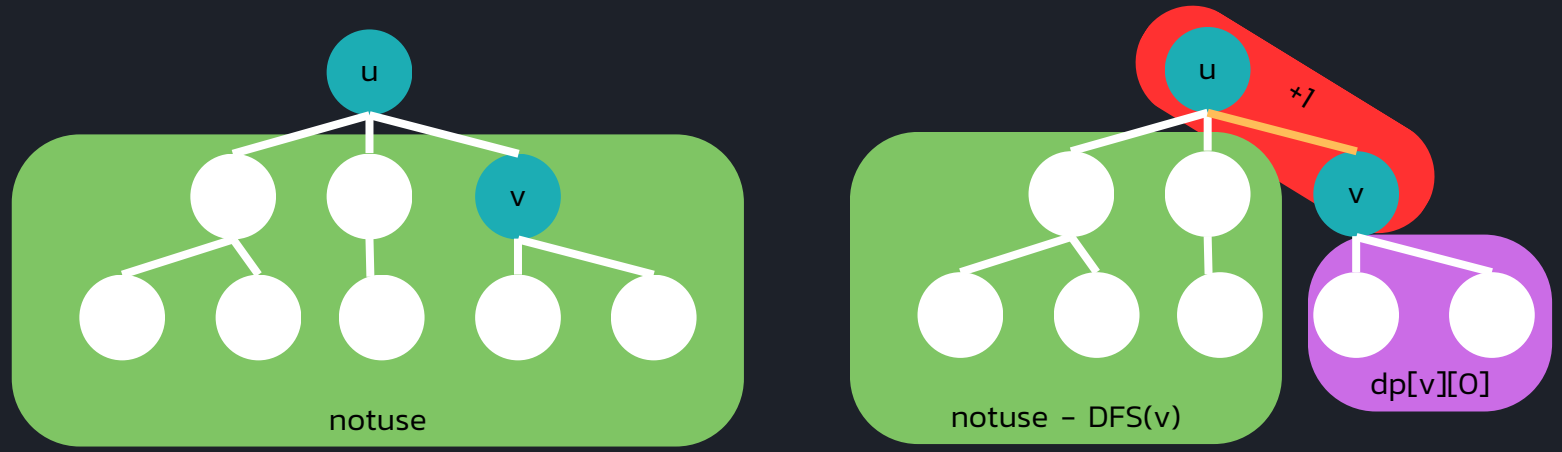
ans = 4

LeViAtHaN

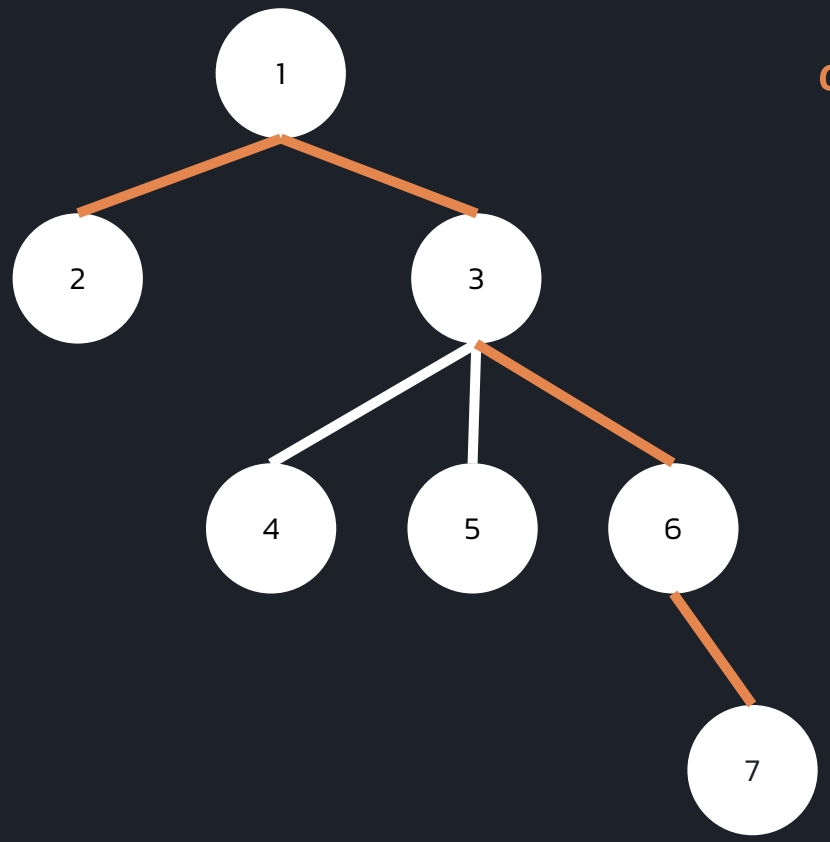
เฉลย CSES TREE MATCHING

เราจะ DP โดยให้ DP[i] return ค่าของจำนวนคู่ที่ node ตัวลูก(รวมทั้งตัวมันเอง) สามารถจับกันได้มากที่สุด
dp[i][0] แทนการไม่เลือก node นั้น, dp[i][1] แทนการเลือก node นั้น,
เมื่อถึง node ใดๆ เรามีสองตัวเลือก คือ

- ไม่เลือก node นั้นให้ไปจับกับ node รอบๆ ใช้ notuse += DFS(ลูก) ได้เลย
- เลือก node นั้น และให้ use = notuse – DFS(v) + dp[v][0] +1



DIAMETER OF TREE



diameter of a tree ระยะห่างระหว่าง node ที่มากที่สุดใน Tree

CSES Tree Diameter

[25 min]

LeViAtHaN

เฉลย CSES TREE DIAMETER

node นี้เป็นได้สองแบบคือ

1.อยู่ใน diameter -> ans = child[left]+child[right]

2.ไม่อยู่ใน diameter

เวลา DFS node จะใช้ pq มาเก็บค่าไว้ก่อน {0,0} แล้ว DFS ลูกรอบๆ
หากมีลูก ก็จะยัดว่าลูกมี path ยาวที่สุดเท่าไหร่(รวม path ที่ชี้เข้ามาใน node นี้แล้ว)มาให้
เราจะ pop ออกมา 2 อันดับแรก ให้เป็น a,b
แล้วเช็คกรณีที่ 1 (อยู่ใน diameter) คำตอบจะเป็น a+b
แต่หากไม่ใช่ ค่า ans ก็จะไม่เปลี่ยนอะไร
สุดท้าย เราก็จะคืนค่า a+1 ให้ node ก่อนหน้าเราไปคำนวนต่อ

เรียงบนต้นไม้ (treeinc)

[20 min]

LeViAtHaN

เฉลย เรียงบนต้นไม้ (TREEINC)

ดูแค่ inc

เราจะ DFS ต่อกับ node ที่น้อยกว่าเท่านั้น ทำให้ทางเดินจะเป็นจาก node ที่น้อยกว่ามาหาเราเสมอ

ดูทั้ง inc/dec

แต่ละ node จะเก็บค่า 2 ค่า ได้แก่ Inc[u]=1 (จากน้อยมา node), Dec[u]=1(จากมากมา node)
และคำตอบจะได้เป็น ans = max (ans, max(max(Inc[u], Dec[u]) , Inc[u]+Dec[u]-1));

REROOTING

ก่อนหน้านี้เราหาคำตอบของ tree ทั้งต้น เช่น หา diameter ที่ยาวที่สุดของ tree
หากเราอยากหาคำตอบที่เฉพาะเจาะจงมากขึ้น อย่าง

- หาระยะห่างรวมทั้งหมดของ node u ไปยัง node ใดๆ
- หาระยะที่ยาวที่สุดจาก node u ไป node ใดๆ

เราจะต้องมีการคำนวนซ้ำ หรือการ rerooting
rerooting ไม่มีสูตรตายตัวและต่างกันไปตามโจทย์ แต่เทคนิคนี้มีจุดร่วมอยู่ เช่น...

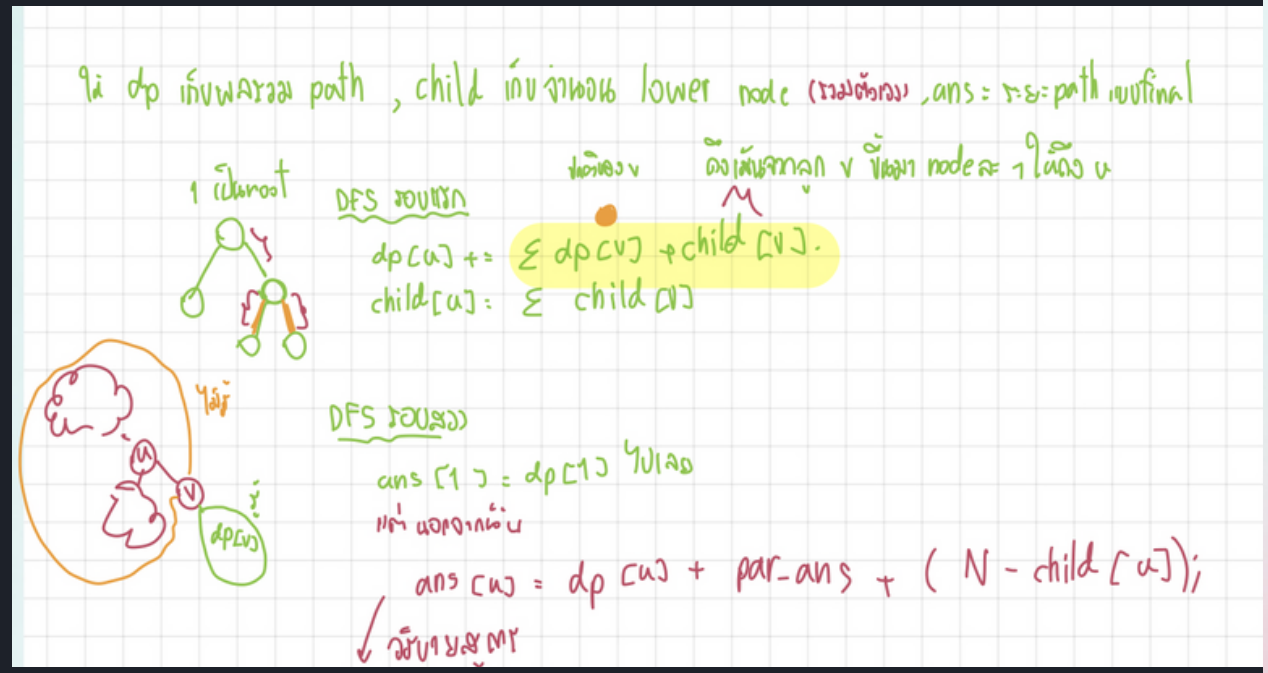
DFS รอบแรก จะทำการเก็บค่า 2 ค่า คือ dp[u] และ child[u]
dp[u] เก็บค่าที่โจทย์สนใจและยังหาได้ในขณะนี้
child[u] เก็บจำนวน node ที่เป็น lower neighbors และตัวเองด้วย เพื่อความสะดวกตอนเพิ่มความยาว path
DFS รอบสอง
ใช้ตัวแปร ans[u] มาหาคำตอบที่แท้จริงของ node นี้

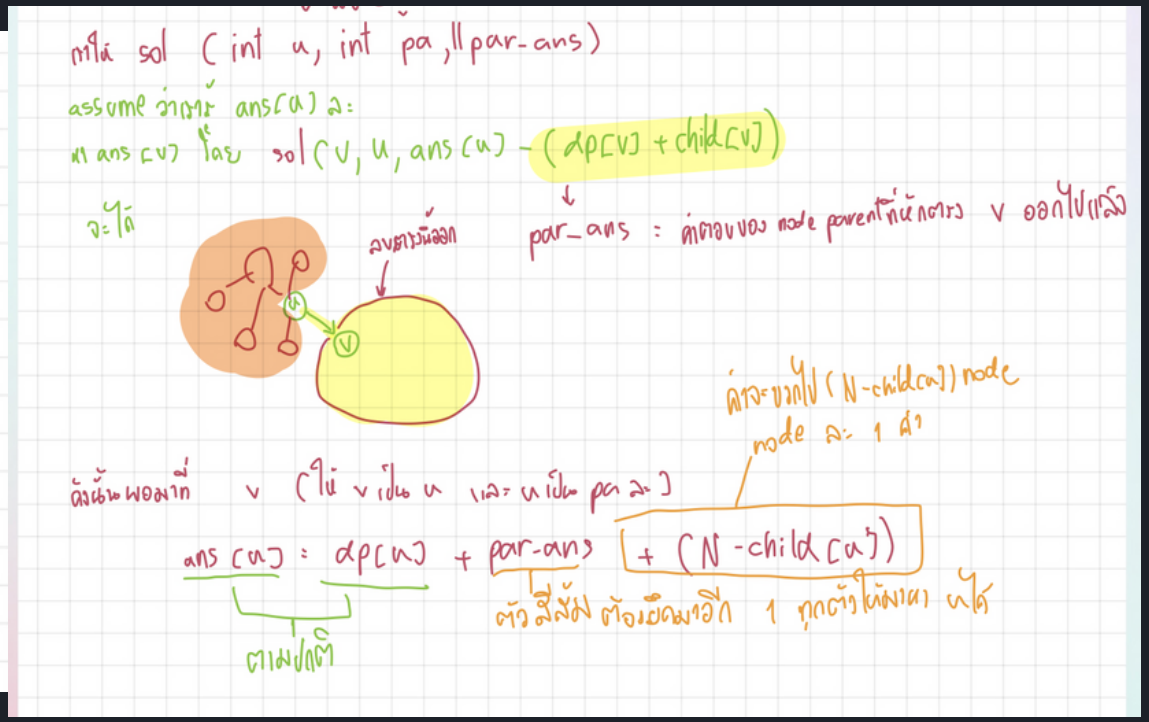
CSES Tree Distances II

[25 min]

LeViAtHaN

เฉลย CSES TREE DISTANCES II





LeViAtHaN

CSES Tree Distances I

[30 min]

LeViAtHaN

เฉลย CSES TREE DISTANCES I

DFS รอบแรก ให้ dp เก็บระยะทางที่ยาวที่สุดจาก u ไปถึงลูกๆได้

DFS รอบสอง เราพิจารณา 3 path คือ

1. path จาก parent
2. path ที่ n<u : prefix[i] แทน path ที่ยาวที่สุดของ node i < u มาถึง u ได้
    a. pf[i]=max(pf[i],pf[i-1]);
3. path ที่ n>u : suffix[i] แทน path ที่ยาวที่สุดของ node i > u มาถึง u ได้
    a. sf[n-i-1]=max(sf[n-i-1],sf[n-i]);

และได้ว่า ans[u]=1+max(patial,(pf.empty()) ? -1 : pf.back());

เพราะ pf.back() คือระยะมากสุดที่ตัวลูกมีให้แล้ว

แล้วใช้

```
op1= (ct==0) ? -1 : pf[ct-1];
op2= (ct==n-1) ? -1 : sf[ct+1];
DFS(v,u,1+max(patial,max(op1,op2)));
```

ไปหาตัวลูกต่อไป

LeViAtHaN

Rose Transportation

[20 min]

LeViAtHaN

เฉลย ROSE TRANSPORTATION

minimum spanning tree โดยใช้ paremetre เป็น w[u]+w[v] < w[d2.u] + w[d2.v]
เพราะ Ti*Dag มองง่ายๆ คือ หากเลือกถนน u->v
ค่า ans += w[u] + w[v] นั้นเอง

HOMEWORK (รอบนี้เฉลยคลิป)

- CSES Sum of Three Values
- CodeForces Books