EDGE DELETION QUESTION

Q) Given an undirected tree with N nodes labeled from 1 to N.

Each node has a certain weight assigned to it given by an integer array A of size N.

You need to delete an edge in such a way that the product b/w the sum of weights of node in one subtree with the sum of weights of nodes in other subtree is **maximized**.

Page No. ______
Date

Return this maximum possible product modulo 10^9+7.

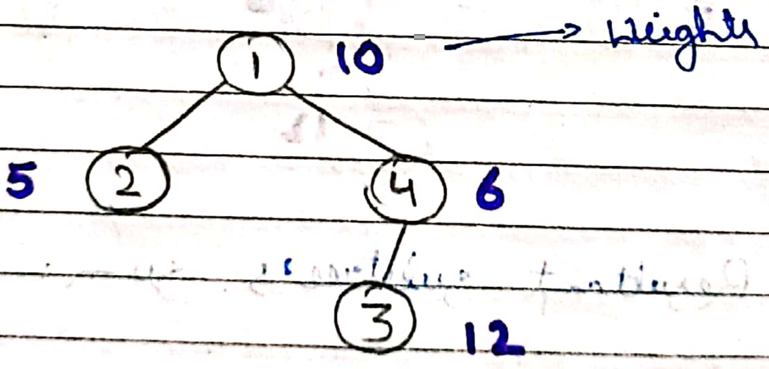
Note:

→ The tree is rooted at a node labeled with 1.

Input:
4 → no. of nodes
A = [10, 5, 12, 6] → weights array
~~4 → no. of nodes~~

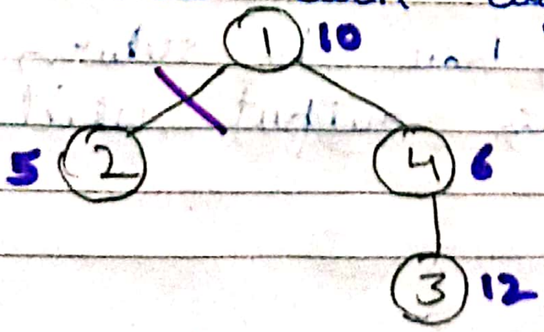
1 2
1 4 } EDGES
4 3

→ This graph will look something like this:



∴ We have 3 edges here.
Let's delete each edge one by one:

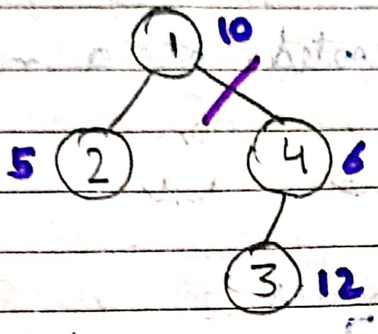
1)



Scanned with CamScanner

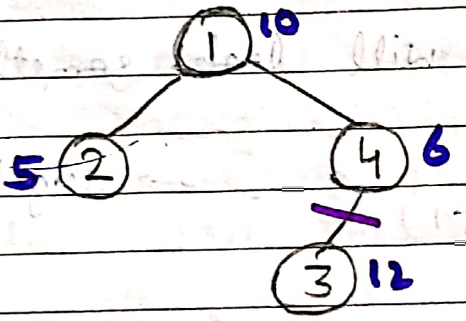
Resultant subtree's sum :

→ 5 and 28
→ Product = 28 × 5 = 140

2) 

Resultant subtree's sum:

→ 15 and 18
→ Product = ~~2~~ 15 × 18 = 270 ✓

3) 

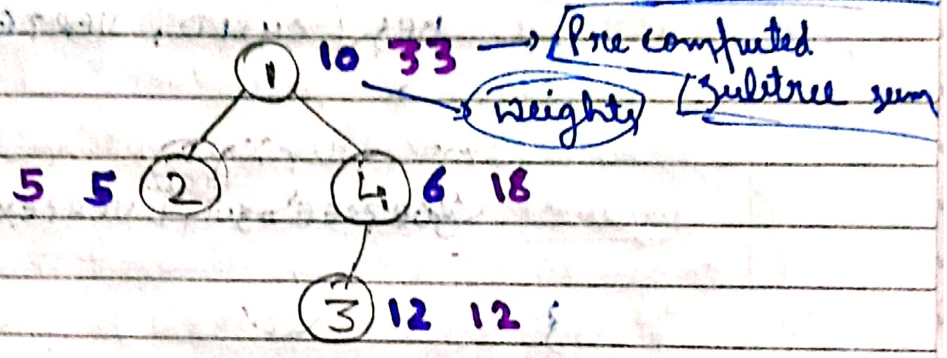
Resultant subtree's sum:

→ 21 and 12
→ Product = 21 × 12 = 252

Finally we have to return maximum product in output which is 270.

Output:
270

APPROACH: Pre compute the subtree sum:
e.g.:



∴ In this let's suppose we ~~delete~~ delete the node b/w ① and ④
∴ Our sums will be:
Precomputed sum of 4 ⇒ 18
" " " 1 − Pre-computed
sum of 4 ⇒ 33−18 ⇒ 15

∴ 18 × 15 = 270

CODE

```
CONST INT M = 1e9+7;
CONST INT N = 1e5 + 10;
VECTOR<INT> G[N];
INT SUBTREE_SUM[N];
-> To store weights of each node
INT WEIGHT[N];

VOID DFS (INT VERTEX, INT PAR = -1)
{
    -> Adding vertex's weight first
```

Page No.
Date

```
        SUBTREE_SUM [VERTEX ]  +=  WEIGHT [VERTEX]
        FOR (INT CHILD : G[VERTEX ])
        {
            IF (CHILD == PAR) CONTINUE;
            DFS (CHILD, VERTEX);

            -> Pre computing subtree sum
            SUBTREE_SUM[VERTEX] += SUBTREE_SUM
                                   [CHILD];
        }
}

INT MAIN( )
{
        INT   n;
        CIN   n;

        -> Taking weights input
        FOR (INT i =1; i <= n; I++ )
        {
            CIN >> WEIGHT[i];
        }

        FOR (INT i=0; i< n-1 ; i++ )
        {
            INT V1, V2;
            CIN >> V1, V2;
            G[V1]. PB (V2);
            G [V2]. PB (V1);
}
```

Scanned with CamScanner

-> Running DFS first to pre-compute subtree sum

```
DFS (1);
```

-> Calculating maximum subtree sum's product.

```
LL ANS = 0;
FOR (INT i = 2; i <= n; i++) // 2 se
start kiya kyunki 2 ke upr edge hai
delete krne ke liye, 1 ke upr nahi hai.
{
    INT PART1 = SUBTREE_SUM[i];
    INT PART2 = SUBTREE_SUM[1] - PART1;
    ANS = MAX (ANS, (PART1 * 1LL * PART2) % M);
}

COUT << ANS << "\n";

RETURN 0;
}
```