1376. Time Needed to Inform All Employees

Medium 181 24 Add to List Share

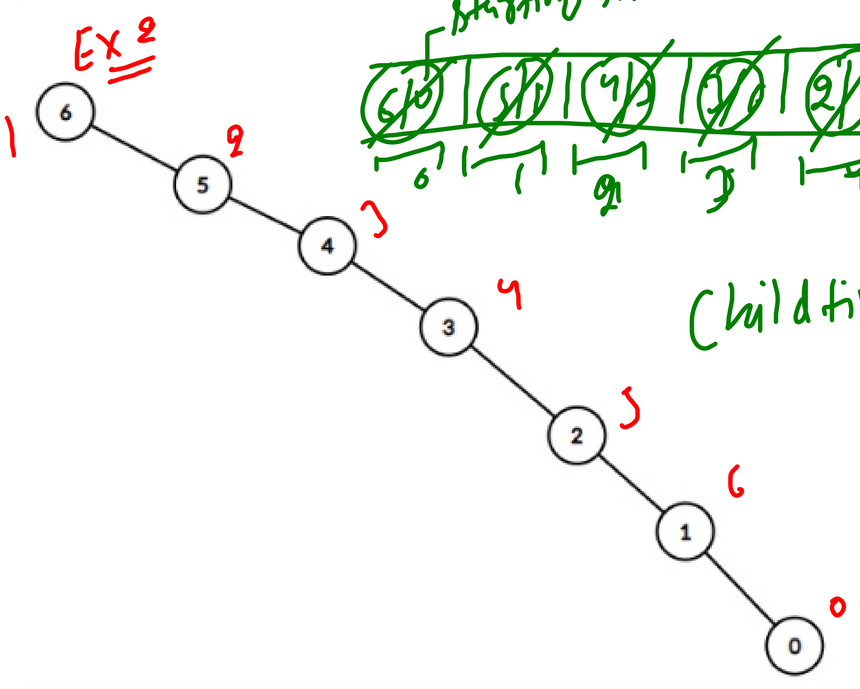
A company has `n` employees with a unique ID for each employee from `0` to `n - 1`. The head of the company has is the one with `headID`.

Each employee has one direct manager given in the `manager` array where `manager[i]` is the direct manager of the `i-th` employee, `manager[headID] = -1`. Also it's guaranteed that the subordination relationships have a tree structure.

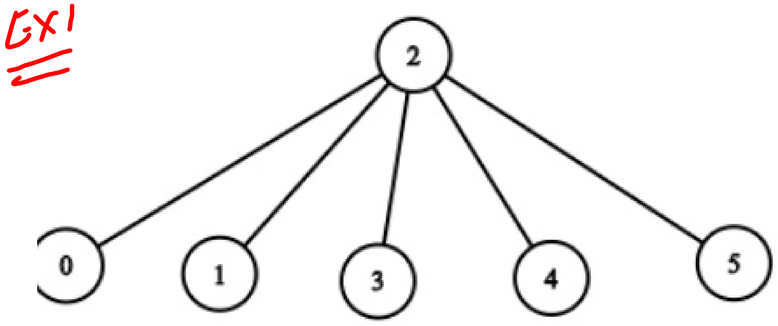
The head of the company wants to inform all the employees of the company of an urgent piece of news. He will inform his direct subordinates and they will inform their subordinates and so on until all employees know about the urgent news.

The `i-th` employee needs `informTime[i]` minutes to inform all of his direct subordinates (i.e After informTime[i] minutes, all his direct subordinates can start spreading the news).

Return *the number of minutes* needed to inform all the employees about the urgent news.

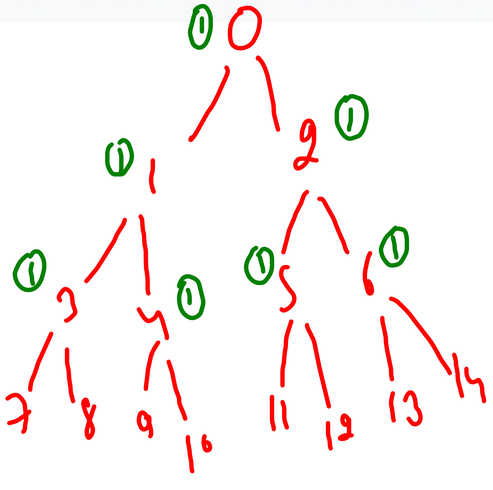


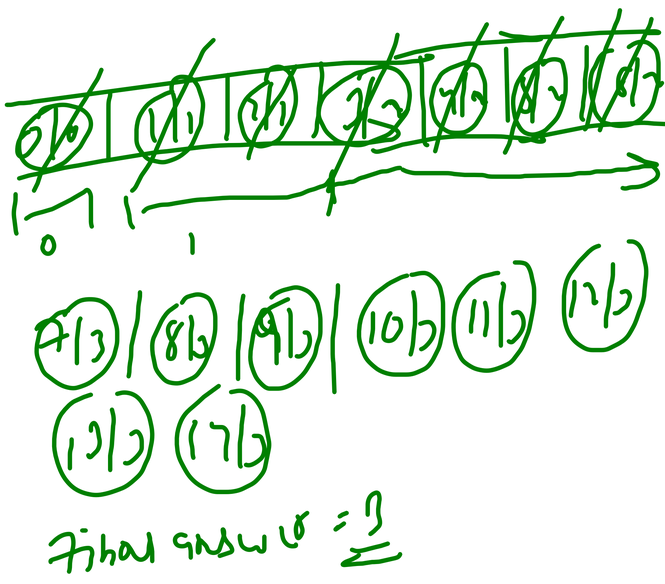

```
Input: n = 7, headID = 6, manager = [1,2,3,4,5,6,-1], informTime =
[0,6,5,4,3,2,1]
Output: 21
```




```
Input: n = 6, headID = 2, manager = [2,2,-1,2,2,2], informTime =
[0,0,1,0,0,0]
Output: 1
Explanation: The head of the company with id = 2 is the direct manager
of all the employees in the company and needs 1 minute to inform them
all.
The tree structure of the employees in the company is shown.
```

```
Input: n = 15, headID = 0, manager = [-1,0,0,1,1,2,2,3,3,4,4,5,5,6,6],
informTime = [1,1,1,1,1,1,1,0,0,0,0,0,0,0,0]
Output: 3
Explanation: The first minute the head will inform employees 1 and 2.
The second minute they will inform employees 3, 4, 5 and 6.
The third minute they will inform the rest of employees.
```

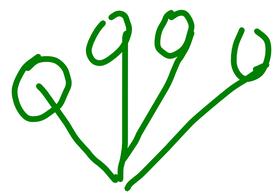




Dfs solution

→ previous solution is Bfs

→ Now Dfs



Logic: harkoi apna inform time lekr
dega. Uske max me ci
apna inform time add
krdo.

```cpp
int dfs(int src, vector<vector<int>> &graph, vector<int> &informTime)
{
    int time = 0;
    for (int e : graph[src])
        time = max(time, dfs(e, graph, informTime) + informTime[src]);

    return time;
}

int numOfMinutes(int n, int headID, vector<int> &manager, vector<int> &informTime)
{
    vector<vector<int>> graph(n);
    for (int i = 0; i < manager.size(); i++)
        if (manager[i] != -1)
            graph[manager[i]].push_back(i);

    return dfs(headID, graph, informTime);
}
```