# Play with Organizational Hierarchy

**Play with Organizational Hierarchy**

**Warning!!!** Read Moodle's A8 PDF for a more complete description. The below description might be incomplete.

You are working in a company with N employees (employee ids range from 0,1, . . . , N−1) that follows a supervision-based hierarchical structure. Apart from having a supervisor, each employee can have at most two other subordinates mentored by him/her (let’s fix an order among the two as: left subordinate and right subordinate). The CEO does not have any supervisor and is also among the N employees. A random list of size N(all are integers) is present which decides the hierarchical structure. For an employee with id i, the id of his/her supervisor is given by the (i+ 1)th number (starts like 1st number, 2nd number,..., so on) in the random list (if CEO’s id is m, there will be a -1 in the (m+ 1)th position in the list). Construct the hierarchical structure which essentially arranges the employees of the company among k levels based on seniority. Assume the CEO will be in level 1, his subordinates will be in level 2, and so on until level k. Perform the following order of STEPs:

1. Print the employee ids starting with the CEO’s, followed by all employees in level 2, level 3,..., level k. For a particular level, print in left to right order.
2. Now for each level starting from level 1, level 2,..., level k print the product of employee ids at that level.
3. Now print list of all N employees by visiting them in the following order of hierarchy:

**Supervisor→Left subordinate→Right subordinate.**

4. Now without affecting the hierarchical structure, increment (don’t decrement!) the values of the employee ids in such a way that for all employees, his/her employee id should be exactly equal to the sum of employee ids of his/her subordinates (NOTE that after this step, the employee ids may not be in the range[0, . . . , N−1] any longer!). HINT: if an employee has no subordinate, no need to change his/her id. But if employee id is already greater than sum of employee ids of subordinates, you need to increment employee id of one of subordinates (let’s fix the left one) as you are not allowed to decrement any value (see Test case2: employee with ids 1 and 0).
5. Now print list of all N employees by visiting them in the following order of hierarchy:

**Left subordinate→Supervisor→Right subordinate.**

6. Lastly, print the value $\max\{\text{hop}_{ij}+1\}$. In terms of the hierarchical structure, $\{\text{hop}_{ij}+1\}$ denotes the length of the path from employee $i$ to employee $j$, given that an employee can only interact with its supervisor or it sub-ordinates.

NOTE: For STEPS 3 and 5, you must use an iterative solution using a stack! it will be checked during design evaluation.

**Input Format**

First line contains: integer $N$ (the number of employees)

Second line contains a list of $N$ space separated integers, determining the hierarchy as described above. (Refer Explanation on Test case in PDF)

**Constraints**

Your code must have the following class:

```
class Employee {
    private:
        int sup_id; // unique obj_id of the supervisor
        int obj_id; // a unique id for the object, that helps maintain and identify the hierarchy (it won't change)
        int obj_val; // the employee id needs to be stored here (it's the data value of the object, that may change). This is required to
be printed/operated on.
        Employee *left_subordinate;
        Employee *right_subordinate;
        // any other data members you require ..
    public:
        // your functions go here ..
};

You are free to add more data members, methods and classes along with aforementioned ones.
```

**Output Format**

– First line contains: integer k (the number of levels in the hierarchy) separated by a space.

– Second line contains N space separated integers denoting the employee ids in the same order as asked in STEP-1.

– Third line should be in the format: < 1 product1 2 product2 . . . k productk > (everything is single space-separated!). Here product k denotes the obtained product for level-k as asked in STEP-2.

– Fourth line contains N space separated integers denoting the employee ids in the same order as asked in STEP-3.

– Fifth line contains N space separated integers denoting the employee ids in the same order as asked in STEP-5.

– Last line contains a single integer denoting the number asked in STEP-6.

**Sample Input 0**

```
7
-1 0 0 2 1 2 1
```

**Sample Output 0**

```
3
0 1 2 4 6 3 5
1 0 2 2 3 360
0 1 4 6 2 3 5
4 10 6 18 3 8 5
5
```

**Sample Input 1**

```
7
1 5 5 2 2 -1 3
```

**Sample Output 1**

```
4
5 1 2 0 3 4 6
1 5 2 2 3 0 4 6
5 1 0 2 3 6 4
1 1 11 6 6 10 4
6
```