COL 106 Assignment 3
Due : Sept 12, 11:59 pm

Storing hierarchical structure of a company

We want to maintain the list of employees in a company. We will be concerned with two quantities associated with each employee in the company -- name of the employee (you can assume no two employees in the company have the same name), and the level of the employee. The level denotes where the person stands in the hierarchy. Level 1 denotes the highest post in the company (say the CEO), level 2 comes below level 1 and so on. There is only 1 person at level 1, but there can be several employees at level i > 1. Each level i employee works under a level i-1 employee, which is his/her immediate boss. Given an employee A, we can form a sequence of employees A',A'', A''', ... where A works under A', A' works under A'', and so on. We say that each employee in A',A'', A''',... is a boss of A. We would like to implement a suitable tree data-structure so that we can implement the following operations :

- **AddEmployee(S,S')** : We want to add a new employee whose name is S. This employee will work under an existing employee S' (note that this automatically decides the level of S, it is one more than that of S').
- **DeleteEmployee(S,S')** : We want to remove an employee whose name is S. S' is the name of another employee in the company who is at the same level as S. When we remove S, all the persons working under S will now work under S'.
- **LowestCommonBoss(S,S')** : S and S' are names of two persons working in the company. This operation prints the name of the employee A who is a boss of both S and S', and among all such persons has the largest level. In other words, we want to find the common boss who is lowest in the hierarchy in the company.
- **Print Employees** : This operation prints the name of the employees in the company. It should print the names according to the levels of the employees. So first it should print the person level 1, then people at level 2, and so on.

Write a program which implements such a data-structure. It should display a menu where one can choose between any of the operations mentioned above. Credit will be given to choice of proper data-structures and efficiency. For example, in the second operation above, one should not just search the entire tree to look for the name of the employee S'. Your program should also catch errors, for example if in the first operation above, there is no employee with name S', then it should say it is an error.