**CSL201: Assignment 3**

**Due Date : September 17, 2013**

**Storing employee records in an hierarchical structure**

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 operations like adding, deletions, printing etc.

Your program should be able to read data from a file. The file contains some integers and some words. The first entry is guaranteed to be an integer. Depending on the value of the integer your program should do the following:

1 Read the name of the CEO. Keep this person at level 1.

2 Read the names in pairs. For each pair (A,B) add the employee A in the tree so that he reports to immediate boss B. Note that automatically level of A will be one more than that of B.

3 Read the name. If the name is “top” ,print the company structure by printing the level number and names of employees at that level in each line, starting with level 1. If another name is given then print the subtree starting with the node containing that name.

4 Read the name K. Print K and within curly braces print the names of all the employees who happen to be boss of K, starting with the immediate boss.

5 Read the name pair (C D). Remove C from the tree and rearrange the employee records such that all persons working under C will now work under D. If no one is reporting to C then simply remove C.

6 Read the name pair (G F). Print names G and F within curly braces. Print the name of the employee who is common boss of G and F at the lowest hierarchy in the company. There will be many bosses of G and F but this boss will be at the highest level amongst all the bosses.

7 print a line of character “-“.

8 Print your name and entry number and exit the program.

Credit will be given to choice of proper data-structures and efficiency. For example, in operation 5 above, one should not just search the entire tree to look for the name of the employee C. Your program should also catch errors, for example if in operation 2 above, there is no employee with name B, then it should say it is an error. Use the notions of exceptions in C++ to implement error checking.

Sample input:

|  |
| --- |
| 1 singh |
| 2 sen singh rao singh pal sen jain sen de rao |
| 3 top 7 4 jain 7 6 dev pal |
| 2 bal rao negi de dev sen |
| 5 sen rao |
| 7 3 rao |
| 7 4 negi |

Sample output:

1 singh

2 sen rao

3 pal jain de

- - - - - - - - - -

jain { sen singh }

- - - - - - - - - -

{ dev pal }sen

- - - - - - - -

2 rao

3 de bal pal jain dev

4 negi

- - - - - - - -

negi { de rao singh }