MAT 2540: Data Structures and Algorithms II

PROJECT 2 (Due 4/5/2015) **Spring 2017** **Professor:** Caner Koca

The goal of this project is to make you become familiar with the trees, and how they are implemented in a computer programming language, like C++. For this assignment, please first see the cpp.sh link on http://canerkoca.com for the **Family Tree Program**, written in C++.

You can complete this assignment in groups of **at most** two people, that means, either alone or with only one of your classmates. You are not allowed to use or copy someone else's code. I will use Blackboard's *SafeAssign* feature to check whether your code is plagiarized from the internet or from your classmate. In case of a plagiarism, the assignment(s) will automatically get a zero, and the case will be brought to the Academic Integrity Committee.

The level of C++ used in this program is very basic, and the program will be explained thoroughly in class. If you prefer to use some other language (like Java, R, C, Python etc.) to implement trees, you are welcome to do so, but you will need to make a presentation to me in my office and explain how your code works.

Assignment:

1. Extend the family tree, by adding 5 more nodes of third generation (5 grand-grandchildren of Adam). It is up to you to decide where to add these nodes, as long as all of them are level 3. Draw a schematic picture of your extended tree, visually, in a program, like Microsoft Word or Paint. Or just draw it on a piece of paper, and take its photo. You will be required to send it to me.

2. Inside the definition of the class “node” in the original code, add an additional new parameter: node* parent. For each node in the tree, define the parents. For example, if the node X is the parent of Y, then you must write **Y.parent=X** in addition to **X.child1=Y**. Thus, every edge in your tree is “doubled”, that is it runs both directions between parent and their child.

3. Similar to the function “**IsChild**” that we defined in the original code, define three other functions of Boolean type, called “**IsParent**”, “**AreCousins**”, “**AreSiblings**”. For example **IsParent(X,Y)** should return true if X is the parent of Y. Similarly **AreCousins(X,Y)** should return true if X and Y are cousins. Note that, in real life, we call two people “cousins”, if they have the SAME grandparents. Save the program on cpp.sh. Create a short URL. The first line of your code must include your name(s) as a comment.

4. In your main function, test all your functions by 3 examples each. For example, if Ben is the child of Adam, your program must print

   ```
   >> Adam is the parent of Ben
   ```

   Or
   ```
   >> Ben and Adam are not cousins
   ```

   Or
   ```
   >> Ben and Adam are not siblings
   ```

   Etc.

5. Think of other uses of the code that you wrote. Can you apply the program in another innovative setting? Or can you extend your program by adding new functions, or classes, or objects?

Grading Scheme:

70% for the accuracy: Your code is expected to do everything what is asked in the assignment. It must be error-free. Expecially if you are working on this assignment in pairs...

20% for the clarity: You are expected to write comments within your code that explain the role of each of the line you add to the original code. Pay attention to spacing and indentation. Name your variables and functions appropriately.

10% for additional work: Think of other possible uses of the code you've written and extend it to make it more useful. Program the extended code, explain what it does, and send the cpp.sh link to me.

Submit the following by e-mail by the due date: Three cpp.sh links containing the code for Assignment, an extended-use-version. Also, send the picture of the extended tree that you used. The due date is sharp, no extension will be given.