University of Newcastle
Discipline of Computer Science and Software Engineering
Semester 2, 2011 - SENG1120/6120

Assignment 3 – Update 2

Due using the Blackboard Turnitin submission facility:
9:00AM – 31 October 2011

NOTE: *The important information about submission and code specifics at the end of this assignment specification*.

Changes:

1. Specification of the procedure to insert elements in the tree
2. Delete elements with grade < 50, instead of <= 50
3. Comments on how to remove elements and keep the binary tree consistency (with reference to textbook)

In lectures we have discussed the use of binary trees. This assignment will require the implementation of a binary tree and several algorithms for element insertion, removal, search and tree traversal.

Assignment Task

Your task in this Assignment is to implement a data structure based on a binary tree, which will store generic objects, i.e. use templates.

The type of object used in this assignment is called `student` and it should store two data values: `string name` and `float grade`.

The data structure should implement functions to add, remove and search for an element, as shown in the lectures.

Specific Steps for Implementation

1) Ask the user for an integer number. That number will be used to initialize C++ random number generator (see commands `srand` and `rand`).
2) Initialize a binary tree and add the following names **in a random order**. The tree should be ordered by name, not grade. As you add the names, assign a random grade, in the range [0,100] to each of them.

Adam	Cameron	Jackson	KiSoon	Nicholas
Adrian	Chris	Jacob	Lance	Ryan
Alexander	Damian	James	Liam	Sang
Andrew	David	Jared	Madison	Shane
Ashley	Dillon	Jodi	Magdalena	Simon
Benjamin	Dylan	Jonathan	Marcus	Thomas
Bradley	Ethan	Joshua	Mark	Timothy
Brobie	Frederik	Julius	Melanie	Trent
Callan	Hong	Kelly	Min	Troy
Callum	Hugh	Kenias	Mitchell	Zaanif

3) Print out all names and grades. You can do that by recursively traversing the tree.
4) Print out how many students had a grade ≥ 85, again, by recursively traversing the tree.
5) Print out the average grade of the class, also by recursively traversing the tree.
6) Search each of the students, one at a time and count the average number of comparisons required to find any given student. That is the summation of all comparisons for all students, divided by the number of students. Print out the result.
7) Delete all students who had a grade < 50. Print out all names and grades of the remaining students; and the average grade of the class. When a node is to be deleted, there are two option:
    a. Left child is empty: Replace the node by its right child.
    b. Left child is not empty: Replace the node by the highest element in the left branch.

    Check the textbook, pages 555-558.
8) Redo Step 6 with the new tree (after the deletion). If a student that is supposed to be there is not found, it means that the tree lost its structural consistency. You did something wrong during the delete operation.

All data types should be implemented as template classes, and use pointers. It should not be necessary for me to note that these classes should be programmed by you, and definitely *not* obtained from any alternate source. You may use hints provided in the lecture slides *only* in writing your code.

Please notice that your code will be tested, so the values on the table should be reproducible when we run your code (i.e. use a seed in the pseudo-random number generator).

Your submission should be made using the Assignments section of the course Blackboard site. **Incorrectly submitted assignments will not be marked. Assignments that do not use the specified class names will not be further marked.** You should provide all your `.h`, `.cpp` and `.template` files and, and a `Makefile`. Also, if necessary, provide a `readme.txt` file containing any instructions for the marker. Each program file should have a proper header section including your name, course and student number, and your code should be properly documented. Finally, a completed Assignment Cover Sheet should accompany your submission. Your assignment should compile and run either in the ES409 computers (the server named **flame**) or using Cygwin. Test it before you submit.

COMPRESS ALL THE FILES INTO A SINGLE .ZIP AND THEN SUBMIT.

This assignment is worth 10 marks of your final result for the course.