Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CS 235 Final Exam

Version 0.18

Instructors: K. Seamons and R. P. Burton

December 10-14, 2012 (Monday through Friday)

due in the Lab on Friday not later than 6:00 p.m.

**Penalty for submitting the Final Exam late: 100%**

Review your posted scores for the semester starting Monday, December 10th, and before requesting a hard copy of the Final Exam

Open Book (course text and your CS 142 course text only), Open Notes (including your own Lab solutions),

Open Secondary Storage Device: yours only

Open Laptop: if you wish

Open Course Website (and CS 142 course website), C++ API (cplusplus.com), and any websites included in the Exam specs, but no other Internet resources (such as non API resources on the C++ website)

Closed Neighbor (and everyone is thy neighbor)

**\*Instructions\***

(Please read carefully)

1. This exam consists of a C++ programming problem. Read and understand the statement of the problem completely before beginning to design, code, and test. As part of your design, consider the test cases that will establish the correctness of your solution. Test your solution thoroughly before submitting it.
2. Produce a solution, which consists of your C++ code, with a comment (at the beginning of the class containing your main method) which includes your name, your student ID number, and “CS 235 Fall 2012 Final Exam.” Save your complete source file(s) using the online submission script. Attribute any code taken from or based on other sources (except for the course texts and the course websites). Attributed code copied from or based heavily on outside sources is worth half credit. Unattributed code copied from or based heavily on outside sources is worth no credit.
3. Understanding the problem correctly is part of the examination. If something seems unclear, ask a CS 235 TA (but no one else) for clarification. You may pose questions to the CS 235 TAs at any time. However, the TAs generally are not permitted to answer questions related to design, C++ implementation, debugging, or testing.
4. Prior to submitting your midterm, score it using the attached scoring sheet (this will help you maximize your points and will help us grade your exam accurately). When you are finished, go to the course website and follow the link labeled “Submit Exam” in the Exam Menu.
5. Sign here (a) to certify that you have reviewed your posted scores for the semester and that you have alerted a TA in writing to any missing or incorrect scores, (b) to request that your final exam be graded, and (c) to certify that no unfair information related to the final exam has been received by you, either directly or indirectly, and that none will be conveyed by you. If we discover that you cheated or assisted someone in cheating, intentionally or unintentionally (including accidentally), your score for this exam may (and probably will) be

rand() % 1

We’re serious.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, 2012

(Signature) (Date)

**Maintaining Balance**

**Background**

Extend the appropriate interfaces to complete the requirements below for Red-Black trees (all files that include the string “RedBlack” in the title). You are free to use the book’s code and the implementations found on the course website. The book’s code appears to be functional for insertion, but may be incomplete, flawed, or missing for removal and printing.

**Part** **1** – Implement insertion, consistent with the techniques presented in the course text (and the PowerPoint slides)

**Part** **2** – Implement removal, consistent with the techniques presented in the course text (and the PowerPoint slides)\*

**Part 3 –** Implement printing by providing a recursive print function that will print a tree of up to height 8 in as illustrated here for a tree of height 3:

Root (value & color)

Left subtree root (value & color)

Left left subtree root (value & color)

Left right subtree root (value & color)

Right subtree root (value & color)

Right left subtree root (value & color)

Right right subtree root (value & color)

**Part 4 –** Use the attached main function which is a driver for the other parts of your program. It is designed to (a) accept a string of zero or more words (consisting of alphabetical characters, no punctuation marks, and with white space as the delimiter separating words) with which it populates your data structure, (b) accept a word which it inserts in your data structure (if it is not already in your data structure), (c) accept a word which it removes from your data structure (if it is present in your data structure), and (d) print out your data structure in the format given above.

\* Red Black tree removal is the subject of Programming Project 6 on pages 687-9. See also the following related Erratum:

<http://bcs.wiley.com/he-bcs/Books?action=resource&bcsId=2892&itemId=0471467553&resourceId=7937>

Extra Credit

Extend the appropriate interfaces to complete the requirements below for 2-3 trees (all files that include the string “TwoThree” in the title).

* (20 points) Satisfy all the requirements (stated above) for 2-3 trees.\*
  + (6 points) insertion
  + (12 point) removal
  + (2) printing

Root (value)

Left subtree root (value)

Left left subtree root (value)

Left middle subtree root (value)

Left right subtree root (value)

Middle subtree root (value)

Middle left subtree root (value)

Middle middle subtree root (value)

Middle right subtree root (value)

Right subtree root (value)

Right left subtree root (value)

Right middle subtree root (value)

Right right subtree root (value)

\* The course text provides a suitable algorithm for insertion into a 2-3 tree, but provides only an illustration of removal from a 2-3 tree.  The following reference provides a detailed description of insertion into and removal from a 2-3 tree, and may be helpful to you:

[http://cs.wellesley.edu/~cs230/fall02/2-3-trees.pdf](http://cs.wellesley.edu/%7Ecs230/fall02/2-3-trees.pdf)

* (25 points) Removal Comparison
  + Make 5, 50, 500, 5000, and 50000 attempts to remove random values (between -500000 and +500000) from 50 or more randomly generated AVL trees and 50 or more Red-Black trees xor 2-3 trees
  + Calculate the average removal times for each quantity of values for AVL trees and for your selected data structure, and record them in the same file
  + Analyze the results and explain your findings in the same file
* (5 points) Identify an insertion or removal case where the selected data structure has a “Big O” which differs from the norm for the selected data structure. Test this case and include in the file an explanation of the behavior.

Implementation Notes for Removal Comparison

* <ctime>

time\_t time(time\_t\* p) returns the number of seconds since January 1, 1970 00:00:00 GMT. If p is not NULL, the return value is stored in the location to which p points

* Each individual insertion or deletion may occur too quickly to measure it. We recommend that you create an array of the values to insert or delete and then measure the time required to insert or delete all the values in the array. Divide the total time by the size of the array to find the average
* Timings are somewhat longer for the first use of a class and function. To keep this from skewing your results, create a dummy tree, add and remove from it once, and then proceed with your timings

**Final Exam Scoring Sheet**

NAME:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_SECTION #:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Day Submitted: \_\_\_\_\_\_\_\_\_\_\_\_ T.A. \_\_\_

Student Grading TA Grading

\_\_\_/ 90 pts \_\_\_/ 90 pts – Required Functionality

\_\_\_/ 40 pts \_\_\_/ 40 pts – Implement Red-Black tree insertion

\_\_\_/ 40 pts \_\_\_/ 40 pts – Implement Red-Black tree removal

\_\_\_/ 10 pts \_\_\_/ 10 pts – Implement Red-Black tree print

\_\_\_/ 10 pts \_\_\_/ 10 pts – Demonstrate good design with your classes and functions

\_\_\_/ 50 pts \_\_\_/ 50 pts – Extra Credit

\_\_\_/6 pts \_\_\_/ 6 pts – Implement2-3 tree insertion

\_\_\_/ 12 pts \_\_\_/ 12 pts – Implement 2-3 tree removal

\_\_\_/ 2 pts \_\_\_/ 2 pts – Implement 2-3 tree print

\_\_\_/ 25 pts \_\_\_/ 25 pts – Compare removal times

\_\_\_/ 5 pts \_\_\_/ 5 pts – Different “Big O”

\_\_\_/ 100 pts \_\_\_/ 100 pts – Total

Student to TA Comments:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

TA to Student Comments:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_