CS 251
Fall 2007
Voicu Popescu
Assignment 5
Due Monday October 1, 11:59PM

Binary trees

1. Implement a C++ binary tree class **BinaryTree** that models arithmetic expressions. An internal node stores an operator represented with a character (i.e. '+', '-', '*', '/') and a leaf stores an integer. It is OK to provide space for both and derive the type of node from the presence or absence of children. Provide the following functionality:
    a. A **constructor** that initializes and builds the binary tree from an arithmetic expression read in from the standard input.
        i. The expression has parentheses isolating all (<operand0> <binary operator> <operand1>) expressions, where <operandi> can be an expression, recursively. For example 1+2+3 is illegal input that you do not have to worry about. The legal form is ((1+2)+3).
        ii. All numbers are integers between 0 and 9.
        iii. Caution: handle space characters correctly. The string defining the expression could or could not have spaces. For example, ((1 + 2)+3) should work just as well as ((1+2)+3).
        iv. Caution: handle negative numbers correctly. For example, (1*-5) is legal input.
        v. You do not have to worry about illegal input.
    b. A **destructor** that frees all dynamically allocated memory.
    c. A method **evaluate** that returns the value of the expression stored in the tree.
    d. A method **draw** that visualizes the binary tree in a text file called `tree.txt', with characters as pixels. An internal node should be drawn with a 5x5 pixels square, with the operator at the center. A leaf should be drawn with a circle 5 pixels in diameter, with the number at the center. The parent-child link should be drawn with a line segment connecting the centers of the two nodes; however, it should stop at the perimeter of the node. A node should be drawn midway between its two children; ignore half pixel issues.
    e. Do not change any of the given function signatures. If you do, you will not receive any credit for that function. The constructor is recursive, the parameter is a point to the parent, and data is read directly from the standard input.
    f. Build the tree structure using the BinaryTree * variables `parent', `left', and `right' which store pointers to the parent, left child, and right child of a node, respectively. Also if the node stores an operator, store it in the variable `oper' and if it stores an integer store it in variable `value'. Also leave these variables publicly accessible. This is so we can test your data structure internally. Failure to follow these rules will result in a 0 for that part of the grading.

2. Extra-credit: Add the member function **isBinaryTree** to BinaryTree that verifies that a tree of nodes given by a pointer to its root is indeed a binary tree. Test that the data-structure is a binary tree by making sure that a non-root node is pointed to by exactly one parent and that a node has 0 or 2 children. Your method will have the following function signature: `bool isBinaryTree() const' and will return `true' if the given node

points to a correct binary tree and `false' otherwise. You must submit a blank file named `extracredit' to have this graded. (2%)

3. Turn in instructions:
    a. Assignment specific:
        i. Turn in files: BinaryTree.h, BinaryTree.cc, main.cc, Makefile, extracredit (if applicable)
        ii. Your code is REQUIRED to compile with /opt/csw/gcc3/bin/g++ (gcc version 3.4.5) on the borg machines. Please update your Makefile or use the one that is provided with the assignment. If your code doesn't compile with this compiler, then even if it compiles with another, your code will still be considered noncompiling.
        iii. Directory structure: all files in the directory lab5
        iv. turnin command executed in the directory containing your lab5 directory: turnin -c cs251 -p lab5 lab5
    b. General:
    See instruction at http://www.cs.purdue.edu/cgvlab/courses/251/As/turnin.html
    Make sure to check it EVERY TIME before submitting a new assignment. It is updated periodically.