Department of Computer Science
PhD Proficiency Exam
Winter 2019
Programming Skills

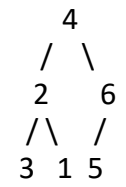
Tree is a hierarchical data structure. Main uses of trees include maintaining hierarchical data, providing access and manipulation operations. Binary trees are special cases of tree where every node has at most two children.

Here, we represent a binary tree using a string consisting of parenthesis and integers. The whole input represents a binary tree. It contains an integer followed by zero, one or two pairs of parentheses. The integer represents the root's value and a pair of parenthesis contains a child binary tree with the same structure. Always start to generate the left child node of the parent first if it exists. If there is only one child node of the parent, you may consider the child node as the left child node of the parent. The further explanation is as follows with two example.

Example 1:
Input : "1(2)(3)"
Tree:

Explanation: first pair of parenthesis contains left subtree and second one contains the right subtree.

Example 2:
Input: "4(2(3)(1))(6(5))"
Tree:

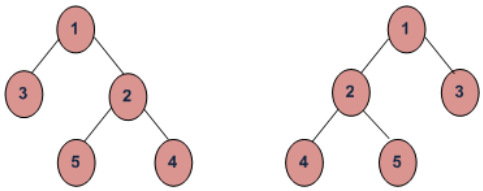


Write a C++ program to complete the following functions or tasks.

1. Define a class, "class bTree", which contains its constructor and destructor as well as the information of nodes in Binary Tree. You can use either a class or struct declaration for nodes in a binary tree. The class bTree must contain functions to insert data into the tree and delete all nodes of the tree. Please feel free to add any member variables and functions which you think necessary for the following functionalities and tasks. (25 points)

2. Construct a binary tree from a string consisting of parenthesis and integers as explained above. Please provide a way to allow users to input a string to construct a corresponding binary tree. You program should print out all of its root-to-leaf paths one per line. (20 points)
For the binary tree in **Example 2**, the output is as follows:
4 2 3
4 2 1
4 6 5

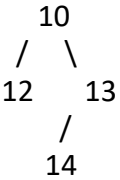
3. Write a member function of class bTree to produce a Mirror of a Tree. The Mirror of a Binary Tree, T, is another Binary Tree M(T) with left and right children of all non-leaf nodes interchanged. Please output the string representation of the Mirror of a given binary tree. (20 points)



Mirror Trees

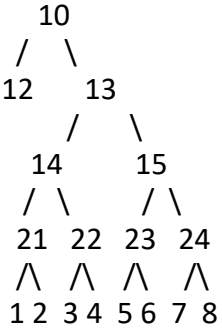
4. Write a member function of class bTree to find the closest leaf to a given node in a binary tree. Given a binary tree and a node x in it, find distance of the closest leaf to x in the binary Tree. If given node itself is a leaf, then distance is 0. See the example below for illustration. (20 points)

Example 3:
Input: The binary tree below and x = pointer to node 13



Output: 1; The distance is 1 as the closest leaf is 14.

Input: The binary tree below and x = pointer to node 13



Output 2; The distance is 2 as the closest leaf is 12 through 10.

5. Write a test program which can take a user's keyboard input of a string to form a binary tree. For example, you program can allow a user to input the string in the above **Example 2** and then define a correct binary tree and print out all of its root-to-leaf paths one per line. Subsequently, the user may request to produce the mirror of the tree; Finally, s/he may request to find the closest leaf to a given node in the binary tree. (15 points)

You can use the above examples for debugging and testing. **Note that, your program must be able to handle any binary trees with any depth (height) and nodes. Your program will be tested and graded using several different binary trees.**

If you believe that certain specification detail is missing, use your best judgment to make an appropriate assumption. Describe your assumption and the decision you made in your comments and in your readme file.

Submission: In the readme.txt, describe your programming environment and details on how to build and execute your program. You may include a makefile. It is your responsibility to ensure that your submission is correct and readable.

No late submissions will be accepted.