Lab 2 Approach Document

# Double-Threaded Binary Tree

# Assignment Objective

# Implement a Double-Threaded Binary Tree and add in-order and reverse-order printing without resorting to recursion

# Assignment Requirements

Your system should have the ability to

* Binary Tree with Two Threads: Create a double-threaded binary search tree (BST) using the included C++ files.
* Management of Node Pointers: In node structures, add thread identification fields. For these fields, implement getter and setter methods.
* Modifications to the method: To support threaded nodes, rewrite the inserthelp() method. For non-recursive printing, modify printhelp() and add the printInorder() and printReverse() methods.
* Documentation and testing: Ensure threading accuracy by testing the tree with the supplied data.

# Approach

* Review all files (documents, code, helps, etc.) provided as part of the assignment
* Select a data structure and implementation that could satisfy this assignment:
  + Create a .cpp file with main() and create a BST object
  + BinNode – what change do we need to make to BinNode relative to threads?
    - Adding setters / getters to BinNode alongside left and right child bit field functions
    - Should be changed so that its pointer type can be determined – is the pointer type a thread or a branch?
  + BSTNode.h
    - BSTNode implements BinNode – what changes are required?
      * Needs to be changed so that it is thread-aware?
      * Should be able to determine which of its pointers is either a branch or a thread
        + Bit fields here

Used to determine whether a pointer is a branch or a thread - getLeftBit() and getRightBit() functions

Setter/getter methods to access and interpret these bit fields – setLeftBit () and setRightBit() functions, getter functionality implemented in getLeftBit() and getRightBit() functions

* + - Add bit or Boolean fields to indicate whether a node pointer is a thread or regular pointer
    - Setter/getter methods to access the bit or Boolean instance variables or modify existing setters/getters as necessary
  + BST.h – Most of the work done here
    - inserthelp() - take advantage of the modified BSTNode in which a pointer can be either a regular pointer or a thread.
      * Gut and recreate method from scratch
      * Use recursion here
      * When I insert a new node, what is the state of the “root” I pass into inserthelp and how does that affect the insert?
      * Is “root” empty, in which case I merely append my new node to the left or right side as appropriate?
      * Is one of the “root” pointers a regular pointer or a thread and is it the side I need to insert on?
    - printhelp() - work with threaded nodes.
      * Uses recursion and threads break its current implementation
      * last task is to modify it so that it will continue to work as designed and ignore threads
    - printInorder() - do an inorder printing of your tree without the use of recursion.
      * Do this next, so that you can print your tree using threads
      * Start at root and use a loop to go as far down the left side of the tree as you can following regular pointers. You are now at the smallest node – print it
      * Now follow the node’s right pointer until you can go left again using the following rules
      * If following a thread don’t go left even if the left subtree is not empty
      * If following a regular pointer go left if the left subtree is not empty
    - printReverse() - do a reverse order printing of your tree without resorting to recursion.
      * rules exact opposite of printInorder()
    - Additional helper methods to make modifying or creating the methods easier
  + Using the following <int, string> values (in the order provided) to build tree: <77, “seventy-seven”>, <70, "seventy">, <75, "seventy-five">, <66, "sixty-six">, <79, "seventy-nine">, <68, "sixty-eight">, <67, "sixty-seven">, <69, "sixty-nine">, <90, "ninety">, <85, "eighty-five">, <83, "eighty-three">, <87, "eighty-seven">, <65, “sixty-five”>

# Build Log

9/25/23 – Today I finished going over all the assignment documentation and other files and have created my approach document up to this point. I created the project folder and imported the required files from Dropbox. Tomorrow I will determine the data structure I plan to use.

9/26/23 – Continued work on the approach document. Meeting with Cameron Kauffman tomorrow to start lab.

9/27/23 – Continued work on the approach document with Cameron.

9/29/23 – Finished work on the approach document with Cameron and started implementation of approach to code. Finished first draft of inserthelp()

9/30/23 – Finished first draft of printHelp() and working on second draft of insertHelp() with Cameron Kauffman

10/1/23 – Spent today debugging with Cameron and trying to get printInOrder() working.

10/3/23 – Finalized project files and submitted project with screenshots.

10/4/23 – Reworking solution without stack implementation with Cameron before resubmission

10/5/23 – Met with Professor Sipantzi and Cameron before resubmission