Milestone 4 – Binary Search Tree, Due April 16, 2025

The following files will be provided to you, for completion of your milestone:

- binary_search_tree.h /* header file containing binary search tree class structure */
- json.hpp // header file for processing json files
- milestone4.h // header file containing main() functions
- tree_node.h // header file containing tree node structure
- tree_node.cpp // tree node constructor
- generate_output.h // header file containing print/write functions
- milestone4.json // json file containing test cases and its transactions
- milestone4_config.json // json configuration (properties) file
- generatedOutputFile.txt // generated output file format (partial results)
- milestone4.cpp /* cpp file containing main, which does the following:
    - Reads configuration file (json format) to:
        - retrieve inputFile (test case file (json format)
        - retrieve outputFile (text file containing generated output)
        - retrieve errorLogFile (text file containing error messages)
    - process inputFile test cases
    - write output to outputFile */

Write a binary search tree implementation, which uses the files listed above, and includes the following in separate cpp files:

- binary_search_tree.cpp – implementation file that contains the following methods:

    1. addToTree - Add a key to the tree
    2. removeNode- Remove a specific key from the BST
    3. getHeightOfTree - Get the height of the tree
    4. getNumberOfTreeNodes - Get the total number of nodes in the tree
    5. contains - Check if a key is in the BST
    6. getRoot - Getter for the root node of the tree
    7. isEmpty - Check if a BST is empty
    8. clear – Removes tree
    9. printNodeFromTree- print the node from the binary search tree
    10. printInorder- print the binary search tree in an in-order traversal
    11. printPreOrder– print the binary search tree in a Pre-order traversal
    12. printPostOrder - print the binary search tree in a Post-order traversal
    13. printDepthFirst - print the binary search tree in a depth-first-order traversal

14. printBreadthFirst- print the binary search tree in a breadth-first-order traversal
15. deleteTree - deletes the tree starting from the specified node
16. getHeight - helper function to calculate the height of a node
17. printInOrderHelper - helper function for recursive in-order traversal
18. printPreOrderHelper - helper function for recursive pre-order traversal
19. printPostOrderHelper - helper function for recursive post-order traversal

The total number of points for this milestone is 90, which will be based upon the following:

- Each submitted/modified file must have student's name (-10% of total milestone points if missing)
- Each submitted/modified file must include description of changes made to a program, and its change date (1)
- Program compiles with all of the provided files (1)
- The following methods run without errors:

    1. addToTree - Add a key to the tree (2)
    2. removeNode- Remove a specific key from the BST (2)
    3. getHeightOfTree - Get the height of the tree (2)
    4. getNumberOfTreeNodes - Get the total number of nodes in the tree (2)
    5. contains - Check if a key is in the BST (2)
    6. getRoot - Getter for the root node of the tree (2)
    7. isEmpty - Check if a BST is empty (2)
    8. clear – Removes tree (2)
    9. printNodeFromTree- print the node from the binary search tree (2)
    10. printInorder- print the binary search tree in an in-order traversal (2)
    11. printPreOrder– print the binary search tree in a Pre-order traversal (2)
    12. printPostOrder - print the binary search tree in a Post-order traversal (2)
    13. printDepthFirst - print the binary search tree in a depth-first-order traversal (2)
    14. printBreadthFirst- print the binary search tree in a breadth-first-order traversal (2)
    15. deleteTree - deletes the tree starting from the specified node (2)
    16. getHeight - helper function to calculate the height of a node (2)
    17. printInOrderHelper - helper function for recursive in-order traversal (2)
    18. printPreOrderHelper - helper function for recursive pre-order traversal (2)
    19. printPostOrderHelper - helper function for recursive post-order traversal (2)

- The following test cases are processed, and produce expected output (10 per test case; 50 total)
- Extra Credit – use industry standard test program and/or extract test cases, in separate json test file

Please accept this GitHub Assignment: https://classroom.github.com/a/OIqJ8pUo