Lab 5: BST

The project **BST** contains the following:

- **Class Node**
  - The **BST** class is included as a **friend class** to be able to access the **private** member variables directly (this will simplify the implementation when using pointers). The class creates objects that store one **int** and two **pointers** to nodes, one to the left and one to the right.
- **Class BST**
  - Creates an object that stores a pointer to the root of the **BST** and a count to keep track of how many nodes are in the tree.
  - **Default Constructor**
    - It initializes the member variable of the class.
  - Function **insert**
    - **Parameter:** An **int** storing an item to insert in the BST.
    - Call the **overloaded function insert** to insert an element in the BST.
  - Function **insert** (**overloaded**)
    - **Parameters:** A **pointer** to a node and a **pointer** to a new node.
      NOTE: The pointer to the node is passed by **reference**.
    - **Recursively** inserts nodes in the BST.
    - This is a **private** function.
  - Function **destroyTree( )**
    - Calls the **overloaded function destroyTree**.
  - **Destructor**
    - Calls the **overloaded function destroyTree**.
  - Function **destroyTree(overloaded)**
    - **Parameter:** A **pointer** to a node.
      NOTE: The pointer to the node is passed by **reference**.
    - Recursively deletes nodes in the BST.
    - This is a **private** function.
- **Text files:**
  - Small list of integers to test the implementation.
    - **data_int_1.txt**
    - **data_int_2.txt**
    - **data_int_3.txt**

- **Main.cpp**
  - **Function processTree**
    - Opens the text files for reading and calls function **insert** of the **BST** class to insert data in the BST.
  - **Function testTree**
    - Tests the implementation of functions that traverse the tree, return the height of the tree, and return the number of nodes in the tree.

Passing a Pointer by Reference

- Why are we passing pointers by reference? Because we are modifying the tree recursively. If we passed the pointers by value, the function will make a copy for each recursion, and when backtracking all the information will be lost (try removing the reference from the insert function). You need to pass a pointer by reference **only** when you are deleting, moving, or inserting nodes (if you are simply traversing, there is no need to pass by reference).
- Why are we creating **private** functions? Because we are passing a parameter, the root, that is a **private** member of the class; therefore, other classes have no access to it and cannot make a function call.

Your job is to complete the implementation of the **BST class** as specified below. Write the declaration of the functions in the **BST.h** class definition, and implement them in the **Functions.cpp** file.

- Function **insert()**
  - This is the **non-recursive** version**.**
  - **Parameters:** The item to insert.
  - **No** duplicates allowed → If the item to insert is already in the tree, output the following message:

    "The item to insert is already in the list – duplicates are not allowed."
  - When testing, make sure you comment out both insert functions that are already provided.

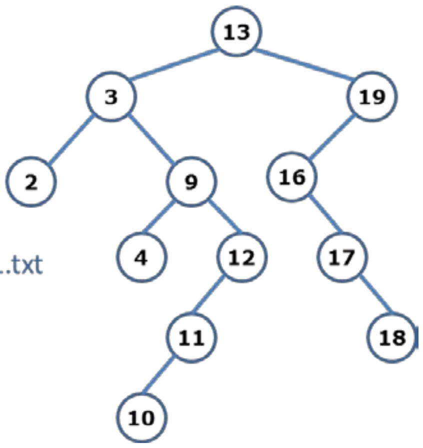
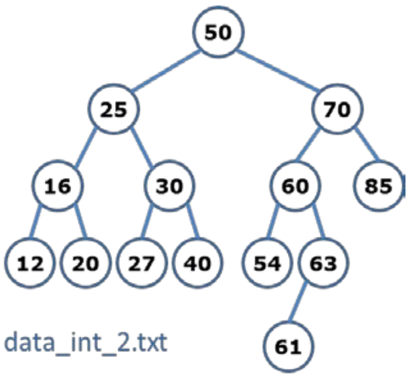
- Functions **preorderTraversal** and **postorderTraversal**:
  - The **overloaded** functions are **recursive** and **private**.
  - Follow the same pattern provided by the **inorderTraversal**, to print out the appropriate sequence.

- Function **totalNodes**
  - Calls the overloaded function totalNodes to return the number of nodes in the BST.

- Function **totalNodes(Node*)**
  - **Parameters:** A pointer to a node.
  - This is a **recursive** function.
  - This is a **private** function.
  - Returns the number of nodes in the tree.

The **Main.cpp** file contains a few testing cases to test your functions. The file reads from the text files.

The **output.txt** file shows the expected output.

 13 3 19 2 9 16 4 12 17 11 18 10 **File:** data_int_1.txt **Nodes:** 12	 50 25 70 16 30 60 85 12 20 27 40 54 63 61 **File:** data_int_2.txt **Nodes:** 14
File: data_int_3.txt has only one item, the root.	