CS 3358 Assignment 4

Due: 11:59pm Tuesday, Nov. 9, 2021

Instructor: Kecheng Yang (yangk@txstate.edu)
Doctoral Instructional Assistant: Muhieddine Shebaro (pri5@txstate.edu)
Grader: Pallavi Krishna Reddy (osg12@txstate.edu)

In this assignment, you are asked to implement several functions in a Binary Search Tree (BST) class, called `myBST`, in `bst.cpp`.

1. Implement the public function `findInBST()`. You can choose to do this using recursion or not. If you choose to use recursion, a helper private function `find_helper()` may be helpful. (non-recursive is the default; if you want to use recursion, you'll need to comment/ uncommented certain parts in the code.)

2. Implement the public function `insertToBST()`. You can choose to do this using recursion or not. If you choose to use recursion, a helper private function `insert_helper()` may be helpful. (non-recursive is the default; if you want to use recursion, you'll need to comment/ uncommented certain parts in the code.)

3. Implement the private functions `preOrder()`, `postOrder()`, and `inOrder()`, which are used to implement public functions `preOrderTraversal()`, `postOrderTraversal()`, and `inOrderTraversal()`, respectively. `preOrder()`, `postOrder()`, and `inOrder()` should be recursive functions, and no loop should be used in them.

Note: If you choose to use recursive implementations, especially for insertion, I would allow you to change the function argument passing from `BinNode*` to `BinNode*&` if this fits your way of the implementation better.
That is, for example, you can replace

```
void insert_helper(BinNode* node, int k)
```

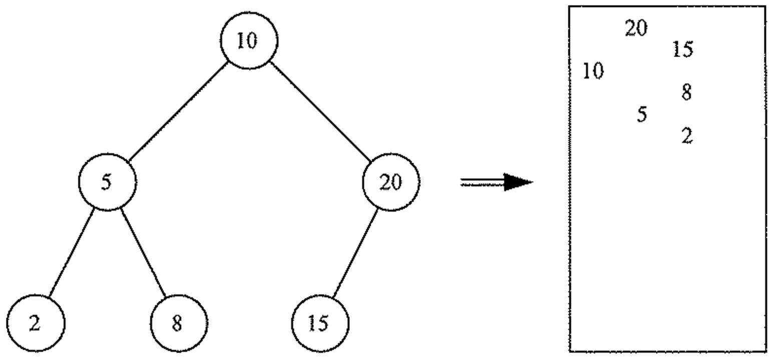
by

```
void insert_helper(BinNode*& node, int k)
```

Nonetheless, just FYI, there is a recursive implementation that doesn't need the '&', as you will see in the sample solution.

Beyond writing codes in this assignment (Just to think about, no submission or grading):

The function `rotatedPrintTree()` prints the BST in a "left-rotated" fashion, i.e., the root on the left and the leaves on the right. For example,



Read carefully the functions for printing the tree (left-rotated), and think about how this function works. Also, think about why we use this function to print the left-rotated tree. Think about writing a function to normally (not rotated) print an arbitrary tree. Which printing is easier to implement?

Submission:
You should submit your work via the assignment tag in the TRACS system.
You should pack `bst.cpp` and an optional README plain text file into a single .zip file to upload to TRACS. The .zip file should be named as `a4_yourNetID.zip`, such as `a4_zz567.zip`

Sample tests:
Note that successes in getting the following test results do not guarantee the correctness of your work and therefore do not guarantee you a satisfactory grade, whereas failures in getting the following test results probably do indicate flaws in your work and you may lose points.

```
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 36
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 20
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 57
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 18
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 44
```

```
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 76
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 93
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 120
Invalid input value (120) !
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: 44
44 is an existing key. No new node has been inserted
Inserting a new node....
Please enter an integer between 0 and 99 as the key, and enter -1
to stop and to see the resulting tree: -1
Print the resulting tree (left-rotated):
                    93
             76
      57
             44
36
      20
             18
preOrderTraversal: 36  20  18  57  44  76  93
postOrderTraversal: 18  20  44  93  76  57  36
inOrderTraversal: 18  20  36  44  57  76  93
Searching a key....
Please enter an integer between 0 and 99 as the key to search,
and enter -1 to stop searching: 57
57 is in this BST.
57 has a left child 44
57 has a right child 76
Searching a key....
Please enter an integer between 0 and 99 as the key to search,
and enter -1 to stop searching: 20
20 is in this BST.
20 has a left child 18
Searching a key....
Please enter an integer between 0 and 99 as the key to search,
and enter -1 to stop searching: 76
76 is in this BST.
76 has a right child 93
```

```
Searching a key....
Please enter an integer between 0 and 99 as the key to search,
and enter -1 to stop searching: 93
93 is in this BST.
Searching a key....
Please enter an integer between 0 and 99 as the key to search,
and enter -1 to stop searching: 55
55 is not in this BST.
Searching a key....
Please enter an integer between 0 and 99 as the key to search,
and enter -1 to stop searching: -1
```