Lab 6

Binary Trees

Build an **Binary Tree** structure that acquires its nodes from a **Stack** of **1000 available tree node elements**.

Provide the following routines, functions or methods:

- A structure or object definition for the Tree Node Element.

- A program that maintains a stack of available tree nodes; performs initialization of the nodes, cleanup of spent nodes, provides routines to allow its caller to acquire and to give back free nodes.

- A program that encapsulates routines necessary to maintain the tree structure:
  -build the root node (a boundary condition),
  -attach a node to the left of its parent,
  -attach a node to the right of its parent,
  -query the contents of a node,
  -query the parent, left and right children(if present) of any node,
  -query the current size of the tree structure.
  -query the level and the degree of a node.

- A MAIN interface that builds the tree by prompting the user for integers. The program accepts the first integer as the contents of the ROOT. Then each subsequent integer is examined as follows. Starting with the root node if the next integer is greater than or equal to the VALUE of that node then you attempt to hang the node as the child to the RIGHT, unless occupied. If less than the VALUE of that node you attempt to hang it to the LEFT, unless occupied. If occupied you must move to this node and designate it as a relative parent node and re-apply the above logic.

- A routine to TRAVERSE the tree in SYMMETRIC INORDER fashion. That is, you visit the left side, then the root, then the right side. You can use either recursive or iterative method to process the tree (and its subtrees).

Testing:
Have a main program that tests each of the above methods:

- Add 20 integer numbers randomly to the structure.
- Show the output of the traversal.

The outputs should look something like:
Construction: Attach node “X” with content of “Y” as the LEFT (or RIGHT) child of “Z”.
The current size of the tree is “N” nodes.
Traversal: Visiting node id “X” which has content “Y”.

Below I have provided as an example the execution output of my program to populate a binary tree using an empty pool of 100 nodes (numbered 99 to 0).

```
1 -- Insert elements into the Tree
2 -- Insert one element onto the Tree
3 -- Query the Root of the Tree
4 -- Query the Current size of the Tree
5 -- Traverse the Tree (preorder)
6 -- Traverse the Tree (inorder)
7 -- Traverse the Tree (postorder)
8 -- Prune the Tree to nothing
9 -- Traverse the Tree in Reverse (inorder)
0 -- Exit
1
Enter an element to add onto the Tree
98

NEW Root node is 99 has contents of 98
The current number of Tree nodes is 1
Enter an element to add onto the Tree
12

Root node is 99 has contents of 98
Attach node 98 with contents 12 as the LEFT child of parent 98
The current number of Tree nodes is 2
Enter an element to add onto the Tree
87

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
                Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Attach node 97 with contents 87 as the RIGHT child of parent 12
The current number of Tree nodes is 3
Enter an element to add onto the Tree
23

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
                Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Move to the RIGHT branch of Node 98 to Node 97
                 Content: 12 to 87
Children of node 98 are:  LEFT child node -1,  RIGHT child node 97
Attach node 96 with contents 23 as the LEFT child of parent 87
```

```
The current number of Tree nodes is 4
Enter an element to add onto the Tree
76

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
                    Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Move to the RIGHT branch of Node 98 to Node 97
                     Content: 12 to 87
Children of node 98 are:  LEFT child node -1,  RIGHT child node 97
Move to the LEFT branch of Node 97 to Node 96
                    Content: 87 to 23
Children of node 97 are LEFT child node 96 RIGHT child node -1
Attach node 95 with contents 76 as the RIGHT child of parent 23
The current number of Tree nodes is 5
Enter an element to add onto the Tree
34

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
                    Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Move to the RIGHT branch of Node 98 to Node 97
                     Content: 12 to 87
Children of node 98 are:  LEFT child node -1,  RIGHT child node 97
Move to the LEFT branch of Node 97 to Node 96
                    Content: 87 to 23
Children of node 97 are LEFT child node 96 RIGHT child node -1
Move to the RIGHT branch of Node 96 to Node 95
                     Content: 23 to 76
Children of node 96 are:  LEFT child node -1,  RIGHT child node 95
Attach node 94 with contents 34 as the LEFT child of parent 76
The current number of Tree nodes is 6
Enter an element to add onto the Tree
76

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
                    Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Move to the RIGHT branch of Node 98 to Node 97
                     Content: 12 to 87
Children of node 98 are:  LEFT child node -1,  RIGHT child node 97
Move to the LEFT branch of Node 97 to Node 96
```

```
            Content: 87 to 23
Children of node 97 are LEFT child node 96 RIGHT child node -1
Move to the RIGHT branch of Node 96 to Node 95
             Content: 23 to 76
Children of node 96 are:  LEFT child node -1,  RIGHT child node 95
Attach node 93 with contents 76 as the RIGHT child of parent 76
The current number of Tree nodes is 7
Enter an element to add onto the Tree
45

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
            Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Move to the RIGHT branch of Node 98 to Node 97
             Content: 12 to 87
Children of node 98 are:  LEFT child node -1,  RIGHT child node 97
Move to the LEFT branch of Node 97 to Node 96
            Content: 87 to 23
Children of node 97 are LEFT child node 96 RIGHT child node -1
Move to the RIGHT branch of Node 96 to Node 95
             Content: 23 to 76
Children of node 96 are:  LEFT child node -1,  RIGHT child node 95
Move to the LEFT branch of Node 95 to Node 94
            Content: 76 to 34
Children of node 95 are LEFT child node 94 RIGHT child node 93
Attach node 92 with contents 45 as the RIGHT child of parent 34
The current number of Tree nodes is 8
Enter an element to add onto the Tree
55

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
            Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Move to the RIGHT branch of Node 98 to Node 97
             Content: 12 to 87
Children of node 98 are:  LEFT child node -1,  RIGHT child node 97
Move to the LEFT branch of Node 97 to Node 96
            Content: 87 to 23
Children of node 97 are LEFT child node 96 RIGHT child node -1
Move to the RIGHT branch of Node 96 to Node 95
             Content: 23 to 76
Children of node 96 are:  LEFT child node -1,  RIGHT child node 95
Move to the LEFT branch of Node 95 to Node 94
```

```
                    Content: 76 to 34
Children of node 95 are LEFT child node 94 RIGHT child node 93
Move to the RIGHT branch of Node 94 to Node 92
                     Content: 34 to 45
Children of node 94 are:  LEFT child node -1,  RIGHT child node 92
Attach node 91 with contents 55 as the RIGHT child of parent 45
The current number of Tree nodes is 9
Enter an element to add onto the Tree
0

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
                    Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Attach node 90 with contents 0 as the LEFT child of parent 12
The current number of Tree nodes is 10
Enter an element to add onto the Tree
10

Root node is 99 has contents of 98
Move to the LEFT branch of Node 99 to Node 98
                    Content: 98 to 12
Children of node 99 are LEFT child node 98 RIGHT child node -1
Move to the LEFT branch of Node 98 to Node 90
                    Content: 12 to 0
Children of node 98 are LEFT child node 90 RIGHT child node 97
Attach node 89 with contents 10 as the RIGHT child of parent 0
The current number of Tree nodes is 11
Enter an element to add onto the Tree
123

Root node is 99 has contents of 98
Attach node 88 with contents 123 as the RIGHT child of parent 98
The current number of Tree nodes is 12
Enter an element to add onto the Tree



1 -- Insert elements into the Tree
2 -- Insert one element onto the Tree
3 -- Query the Root of the Tree
4 -- Query the Current size of the Tree
5 -- Traverse the Tree (preorder)
6 -- Traverse the Tree (inorder)
7 -- Traverse the Tree (postorder)
```

```
8 -- Prune the Tree to nothing
9 -- Traverse the Tree in Reverse (inorder)
0 -- Exit


6
Traversal Inorder
Visit node: 90 with contents 0
Visit node: 89 with contents 10
Visit node: 98 with contents 12
Visit node: 96 with contents 23
Visit node: 94 with contents 34
Visit node: 92 with contents 45
Visit node: 91 with contents 55
Visit node: 95 with contents 76
Visit node: 93 with contents 76
Visit node: 97 with contents 87
Visit node: 99 with contents 98
Visit node: 88 with contents 123
```