GitHub: https://github.com/tudor-alexa99/LFTC

Symbol table implemented using a Binary Search Tree

Identifiers and constants share the same ST

Class Node:

- *Contains pointers to the left and right child nodes*
- *Contains a field for data*

Class BST:

- *Contains a root Node*

Insert(String data):

If there is a node in the tree that contains the same data value, return the node. Else, it goes down the tree until it finds a leaf node and saves the data in a new Node. Returns the newly created Node.

Search(String data):

Performs a lookup operation for the data in the tree. If such a value is found, returns the node containing it. Else, returns NULL.

Class SymbolTable:

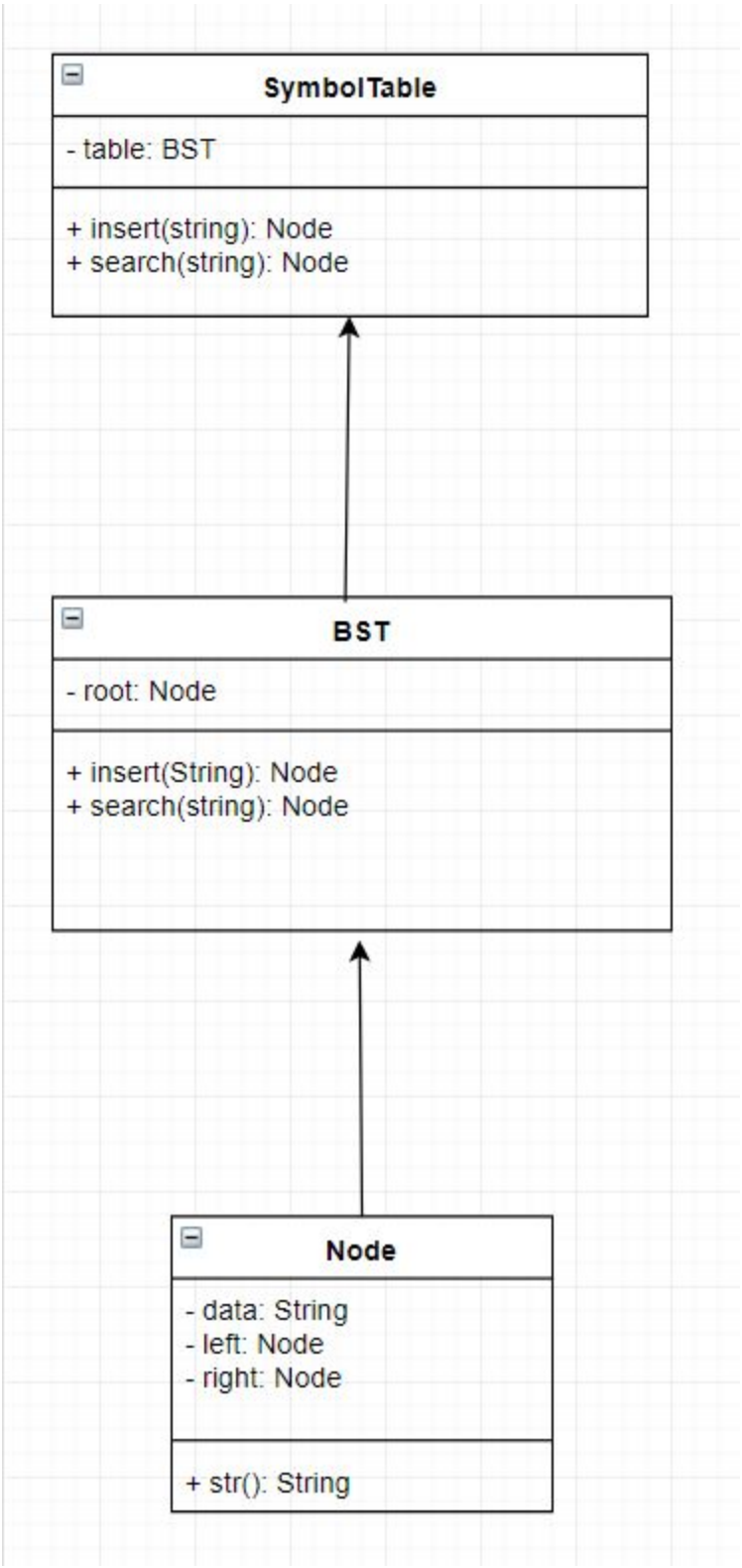
- *Contains a BST*

Insert(String value):

Adds an identifier or a constant to the BST. If the value inserted already exists, returns the Node containing it. Else, creates a new Node with the value and returns it.

Search(String value):

Performs a lookup operation for a constant or an identifier in the ST. If the value is found, returns the Node containing it. Else, return NULL.


SymbolTable
- table: BST
+ insert(string): Node
+ search(string): Node
BST
- root: Node
+ insert(String): Node
+ search(string): Node
Node
- data: String
- left: Node
- right: Node
+ str(): String