Github Link – Geanovu Medeea-Elena 933/1

## Symbol Table Implementation as an Alphabetically Sorted Binary Search Tree

The SymbolTable class represents a data structure that stores symbols in a Binary Search Tree (BST), where each symbol is paired with a unique integer code. The tree keeps symbols in alphabetical order, with methods for adding, finding and printing the contents. The symbols are assigned codes starting from 0, and the code increases for each new entry. As the solution uses a unique for identifiers and constants Symbol Table, all entries are stored in the same SymbolTable instance.

The Node class represents the elements of the tree, a pair of a position and a symbol.

-A symbol, stored as a generic type.

-A code, a unique number assigned to each symbol.

-Links to left and right child nodes, which represent other symbols and their codes stored in the tree.

### Methods of the SymbolTable:

int insert(T key) - adds a new symbol to the table. If the symbol already exists, it won't add it again.

-Calls insertHelper() to place the symbol in the correct spot in the tree.

-Returns the unique code associated with the symbol (either the new code or the one that was previously assigned if it already exists).

Node<T> insertHelper(T symbol, Node<T> currentNode) - returns the newly added node(currentNode).

-If the current node is null, it creates a new node with the given symbol and its unique code.

-It compares the new symbol with the symbol in the current node: If the new symbol comes before the current node's symbol (in alphabetical order), it tries to insert it in the left subtree, otherwise in the right tree.

int find(T symbol - searches for a symbol in the tree and returns its code.

- The helper function findHelper() locates and returns the position of a symbol, or -1 if the symbol doesn't exist.

Node<T> findHelper(T symbol, Node<T> currentNode) - searches the tree for the specified symbol by comparing it alphabetically with the symbols in the nodes.

-It moves to the left or right depending on where the symbol should be, based on alphabetical order.

-Returns the node with the symbol if found, or null otherwise.

void inOrder(Node<T> currentNode) - prints all the symbols and their codes in alphabetical order using in-order traversal.