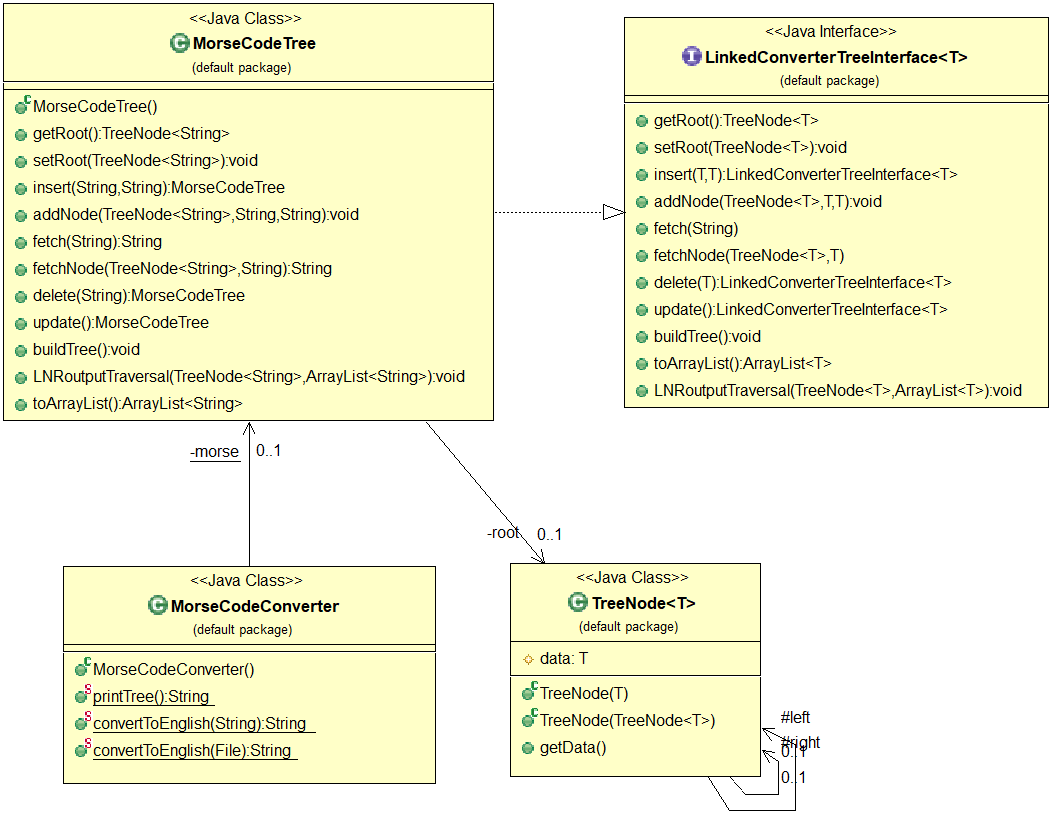
**CMSC 204 Assignment 5**

**Write-up Document**

**Submitted by: Rabindra Raj Suwal**

**UML Diagram**



**Pseudocode**

**TreeNode:**

* Create a public TreeNode(T dataNode) constructor
* set left and right child to null and data to the dataNode
* create another constructor TreeNode(TreeNode<T> node) to make deep copies of the node
* create a public getData() method to return the data in TreeNode

**MorseCodeTree:**

* implements LinkedConverterTreeInterface
* Create a root
* Create a constructor MorseCodeTree() which calls buildTree() method.
* Create getRoot() method which returns reference of the root
* Create setRoot() method to set root of MorseCodeTree
* Create MorseCodeTree insert method to add elements in the tree
  + This method calls addNode()method
* Create addNode() method that adds to the tree
* Create fetch() method to fetch elements from the tree
  + This method calls fetchNode()method
* Create fetchNode() method that fetches elements from the tree
  + ‘.’ (dot) means to traverse left
  + ‘-‘ (dash) means to traverse right
* Create buildTree() method that builds MorseCodeTree by inserting nodes in specified places
* Create toArray() method that returns array list of items in linked tree
* Create LNRoutputTraversal() method that puts the contents of the tree in an ArrayList in LNR (Inorder)

**MorseCodeConverter:**

* Make a static object of MorseCodeTree which calls the MorseCodeTree constructor
* Create a printTree method that returns a string with all the data in the tree in LNR order with a space in between them.
* Create two static methods convertToEnglish
  + Pass a string object (morse code) in one method and a file containing morse codes in the other
  + These static methods use the MorseCodeTree to convert from morse code to English characters.

**Test Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Input** | **Actual Input** | **Expected Output** | **Actual Output** | **Did the test pass?** |
| 1 | .... .. / .. / .- -- / .-. .- -... .. -. -.. .-. .- | .... .. / .. / .- -- / .-. .- -... .. -. -.. .-. .- | Hi I am Rabindra | hi I am rabindra | Yes |
| 2 | .... . .-.. .-.. --- / .-- --- .-. .-.. -.. / -. .. -.-. . / - --- / -- . . - / -.-- --- ..- | .... . .-.. .-.. --- / .-- --- .-. .-.. -.. / -. .. -.-. . / - --- / -- . . - / -.-- --- ..- | Hello world nice to meet you | hello world nice to meet you | Yes |
| 3 | -- --- -. - --. --- -- . .-. -.-- / -.-. --- .-.. .-.. . --. . | -- --- -. - --. --- -- . .-. -.-- / -.-. --- .-.. .-.. . --. . | Montgomery college | montgomery college | Yes |

**Learning Experience**

In this assignment, we had to convert Morse code into its English Translation. This assignment required the concepts of generic classes, utility class (all static methods), linked trees and building a tree for conversion purposes. We were also required to make out own Junit test classes. We were also provided the driver class, Junit tests and Javadoc for the class we had to make.

In this assignment I got more practice on recursive methods. The LNR Traversal recursive method was the most challenging. I had to perform traversals to check if a node had any children until I reached a leaf node, then move back up adding all the elements in the arraylist. The addNode and fetchNode methods were comparatively easier as I did not have to store each nodes that were visited. Another problem I faced during this assignment was with convert methods in the converter. I was getting an empty string everytime I called the fetch method instead of English letters that corresponded to the morse code. I stored each string that fetchNode method returned when it was called recursively to fix the problem.

Other than these two concerns, this assignment was very interesting to learn about Morse code. The javadocs and sample Junit helped a lot as always.