Project 5

Design

Data Element – TreeNode class

I presumed tree node would be the easiest class to implement due to the simplicity of a node class. First, I created all the data fields of type TreeNode, left – right. Data would hold any type by utilizing generic as a place holder. Two constructors were made one taking in data and the other taking in a TreeNode<T> node. The first initializes the “left” and “right” nodes to null, the other gets the data from incoming node and sets left and right of that node to our data fields. GetData returns the nodes data for any TreeNode<T>, method is generic.

Data Structure – MorseCodeTree class + Utility class – MorseCodeConverter class

Reading over Javadoc was scary at first. There were numerous methods needed implementation which gave sorted fear factor. But I just followed the doc and Javadoc given for each method. I did some pseudo code initially to get a rough sketch and overall idea.

Test Case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prompt | Input | Output | Excepted Output | Screenshot |
| Insert string code | .. / .-- .- -. - / .- / .--. ..- .--. .--. -.-- | i want puppy | i want puppy |  |
| Add invalid morse code | hello | hello is printed in console | hello is printed in console | Graphical user interface, text, application, chat or text message  Description automatically generated |
| Read file | monkey | Monkey see monkey do | Monkey see monkey do | Graphical user interface, text, application, email  Description automatically generated |
| Add random morse code | . . . -. -.-. .-- | eeencw | eeencw | Graphical user interface, text, application  Description automatically generated |

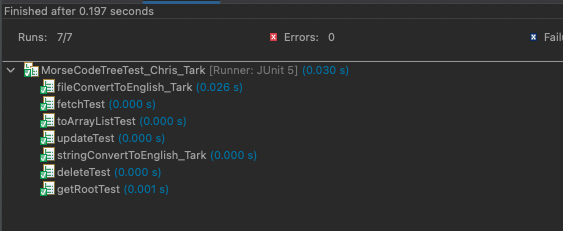
Junit

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text

Description automatically generated



Learning Experience

Another thought-provoking project to work upon. Project 5 used generics, linked trees, topics that were not initially worked thoroughly were static methods and building a tree for conversion purposes. I enjoyed developing a process for converting morse code into the English language. Recursion is what made all the parts move orderly but gave me the biggest problems. I have learned to minimize as many data fields as possible in a recursive method because returning data can get very complicated. I aim for a base class and any other if statement needs a return. Wording with strings is tricky especially targeting specific portions of a whole string. I find it best to us the split method to arrange each section into an array element.