Assignment 5

CS204

Creating a Tree Class, Building a Tree

Converting Morse code to English characters

Concepts tested by this program:

Generic Classes

Utility Class (all static methods)

New concepts tested by this program

Linked Trees

Building a Tree for conversion purposes

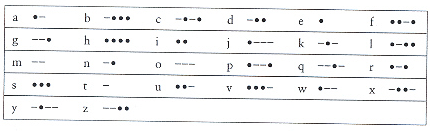
Samuel F. B. Morse produced the first working telegraph set in 1836. This made transmission possible over any distance. The first Morse Code message, "What hath God wrought?", was sent from Washington to Baltimore.

Morse code was extensively used for early [radio](http://en.wikipedia.org/wiki/Radio) communication beginning in the 1890s.

In the early part of the twentieth century, the majority of high-speed international communication

was conducted in Morse code, using telegraph lines, undersea cables, and radio circuits.

Morse code can also be transmitted using light which sometimes happens between ships at sea. It is used in emergencies to transmit distress signals when no other form of communication is available. The standard international distress signal is •••---••• (SOS).



Your assignment is to write a generic TreeNode class, a MorseCodeTree class and a MorseCodeConverter class. There is no GUI required for this assignment. Your classes will be tested with Junit tests.

**TreeNode class**

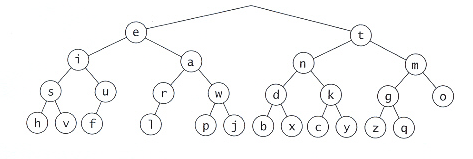
This generic class is used in the MorseCodeTree classes. The class consists of a reference to the data and a reference to the left and right child. Follow the Javadoc that is provided. The Javadoc only lists those public methods that are required to pass the Junit tests. You may add any private methods you need for your design.

**MorseCodeTree class**

A generic linked binary tree which inherits from the LinkedConverterTreeInterface. The class uses an external generic TreeNode class parameterized as a String: TreeNode<String>. This class uses the private member of root. Nodes are added based on their morse code value. A ‘.’ (dot) means to traverse left and a ‘-‘ (dash) means to traverse right. The constructor will “build the tree”. Follow the Javadoc that is provided. The Javadoc only lists those public methods that are required to pass the Junit tests. You may add any private methods you need for your design.

**Building the Data Structure (buildTree)**

Your MorseCodeTree is a tree 4 levels. Insert a mapping for every letter of the alphabet into the tree map. The root is a TreeNode with an empty string. The left node at level 1 stores letter ‘e’ (code ‘.’) and the right node stores letter ‘t’ (code ‘-‘). The 4 nodes at level 2 are ‘i’, ‘a’, ‘n’, ‘m’ (code ‘..’, ‘.-‘, ‘-.’, ‘—‘). **Insert** into the tree by tree level from left to right. A ‘.’ will take the branch to the left and a ‘-‘ will take the branch to the right. This is the structure of the tree.



**Using the Data Structure**

Use the MorseCodeTree to convert Morse Code to English by taking the code and finding it’s corresponding English letter by traversing the MorseCodeTree, ‘.’ branches to the left and ‘-‘ branches to the right. The code ‘.--.’ would branch to the left, then to the right, then to the right, then to the left to **Fetch** the letter ‘p’. Each letter is delimited by a space (‘ ‘). Each word is delimited by a ‘/’.

**MorseCodeConverter – Utility Class**

The MorseCodeConverter contains a static MorseCodeTree object and constructs (calls the constructor for) the MorseCodeTree.

This class has two static methods *convertToEnglish* to convert from morse code to English. One method is passed a string object (“.-.. --- ...- . / .-.. --- --- -.- ...”). The other method is passed a file to be converted. These static methods use the MorseCodeTree to convert from morse code to English characters. Each method returns a string object of English characters.

There is also a static printTree method that is used for testing purposed – to make sure the tree for MorseCodeTree was built properly.

Use the Javadoc provided to make sure that your MorseCodeConverter class follows the method headers so that the MorseCodeConverterTest will run correctly.

**The JUnit Test Class**

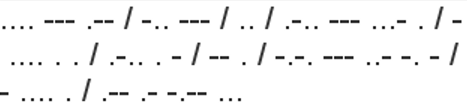
You must add at least 1 test for MorseCodeConverter.convertToEnglish(String) and at least 1 test for MorseCodeConverter.convertToEnglish(File) to the MorseCodeConverterTest class. Include your test file with your code files.

**Test Cases:**

Hello World



How do I love thee let me count the ways



Some suggestions:

1. There is a morse code translator at:

<http://morsecode.scphillips.com/jtranslator.html>

it will help you make test cases

**Program Grade Sheet**

Assignment #5

CS204

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Blackboard Date/Time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## DOCUMENTATION

# CheckList for Assignment 5 is included and completed 1 pt \_\_\_\_\_

# Class documentation - Javadoc for all user created classes 4 pt \_\_\_\_\_

Class Description, @author

Method description, @param, @return

Test Cases 6 pt \_\_\_\_\_

JUnit Test Class

Implement STUDENT test for the MorseCodeConverter

Create a test for your MorseCodeTree class

UML Diagram 4 pt \_\_\_\_\_

Lessons Learned 5 pt \_\_\_\_\_

In 3+ paragraphs, highlight your lessons learned and learning experience from working on this project. How did you do? What have you learned? What did you struggle with? How will you approach your next project differently?

## PROGRAMMING

Programming Style

Internal class documentation (within source code) 5 pt \_\_\_\_\_

Class description using Javadoc

Author’s Name, Class, Class Time, @author

Methods commented using Javadoc, @param, @return

Accuracy - Received correct output

Public test given you 6 pt

Your tests for MorseCodeConverter and MorseCodeTree 4 pt \_\_\_\_\_

Private tests 10 pt \_\_\_\_\_

Program Details

1. MorseCodeConverter 20 pt \_\_\_\_\_
2. Uses the MorseCodeTree
3. Static method convertToEnglish that takes a String object of morse code

and returns a String object of corresponding English characters

1. Static method convertToEnglish that takes a File object of morse code

and returns a String object of corresponding English characters

1. Static method to print the tree
2. Follows the Javadoc provided
3. MorrisCodeTree 25 pt \_\_\_\_\_
4. Implements the LinkedConverterTreeInterface
5. Creates and builds the MorseCodeTree
6. Uses recursive method to add a node to the tree
7. Uses recursive method to fetch a node from the tree
8. Follow the Javadoc provided
9. TreeNode 10 pt \_\_\_\_\_
   1. Follow the Javadoc provided

Total 100 pt \_\_\_\_\_

Comments: