CMSC204
Assignment #5

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 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).

Assignment Description

Write the classes required to create a Morse Code Converter Utility. Your Morse Code Converter Utility will be using a generic linked binary tree with generic TreeNodes to convert Morse Code into English. There is no GUI requirement for this assignment. You are supplied a GUI for testing purposes.

Concepts tested by this assignment

Generic Classes
Utility Class (all static methods)
Linked Trees
Building a Tree for conversion purposes

Classes

Data Element - **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.

Data Structure - **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 call the method to "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.

Utility class - MorseCodeConverter

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 purposes – to make sure the tree for MorseCodeTree was built properly.

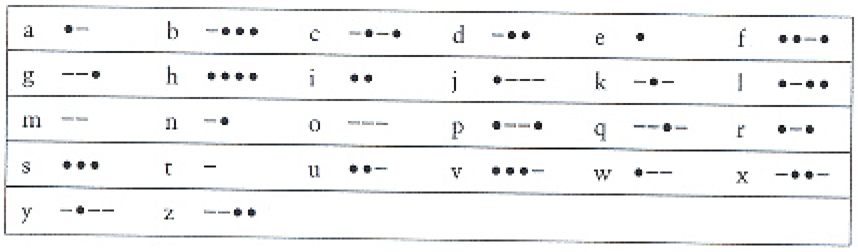
Use the Javadoc provided to make sure that your MorseCodeConverter class follows the method headers so that the MorseCodeConverterTest will run correctly.

Testing - JUnit Test Classes

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

Assignment Details

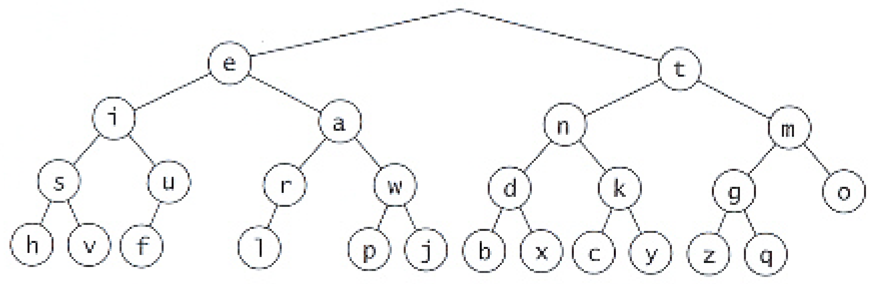
This is a table for the conversion from Morse Code to alpha letters.



a	•–	b	–•••	c	–•–•	d	–••	e	•	f	••–•
g	––•	h	••••	i	••	j	•–––	k	–•–	l	•–••
m	––	n	–•	o	–––	p	•––•	q	––•–	r	•–•
s	•••	t	–	u	••–	v	•••–	w	•––	x	–••–
y	–•––	z	––••								

Building the MorseCodeTree (method buildTree)

Your MorseCodeTree is a 4 levels tree. 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 MorseCodeTree

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 '/'.

Some suggestions:

1. There is a morse code translator at:

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

This will help you build files and test cases for your JUnit Tests.

Examples

Test Cases:

Hello World

.... . .-.. .-.. --- / .-- --- .-. .-.. -..

How do I love thee let me count the ways

.... --- .-- / -.. --- / .. / .-.. --- ...- . / -
.... . . / .-.. . - / -- . / -.-. --- ..- -. - /
- .... . / .-- .- -.-- ...

Deliverables

Deliverables / Submissions:
Design: UML class diagram with algorithm (pseudo-code) for methods

Implementation: Submit a compressed file containing the follow (see below): The Java application (it must compile and run correctly); Javadoc files in a directory; a write-up as specified below. Be sure to review the provided project rubric to understand project expectations. The write-up will include:

- UML diagram
- In three or more paragraphs, highlights of your learning experience

Deliverable format: The above deliverables will be packaged as follows. Two compressed files in the following formats:

- LastNameFirstName_Assignment4_Complete.zip, a compressed file in the zip format, with the following:
    - Write up (Word document) - reflection paragraphs
    - UML Diagram - latest version (Word or jpg document)
    - doc (directory) - Javadoc
        - File1.html (example)
        - File2.html (example)
    - src (directory)
        - File1.java (example)
        - File2.java (example)

- LastNameFirstName_Assignment4_Moss.zip, a compressed file containing one or more Java files:
    - File1.java (example)
    - File2.java (example)

  This folder should contain Java source files only