Morse Code Translator

Project 3B

3 March 2021 - Intro to Data Structures with Java

Github Link: https://github.com/rdmallinson7/Project_3_DS.git

ISSUED BY

Dayu Wang - St. Charles Community College

TEAM SIX

Rachael Mallinson

Sofia Nikas

Aleksis Martin

Design Explanation

Data Structures utilized: Binary Tree

We build the binary tree from an input file with letters from the English alphabet and their Morse code translations. To build the tree, we trace the path from the root to the letter's place in the tree by reading each character in the morse code and branching left for a dot and right for a dash, creating new nodes along the way to keep the connection from the root to each leaf.

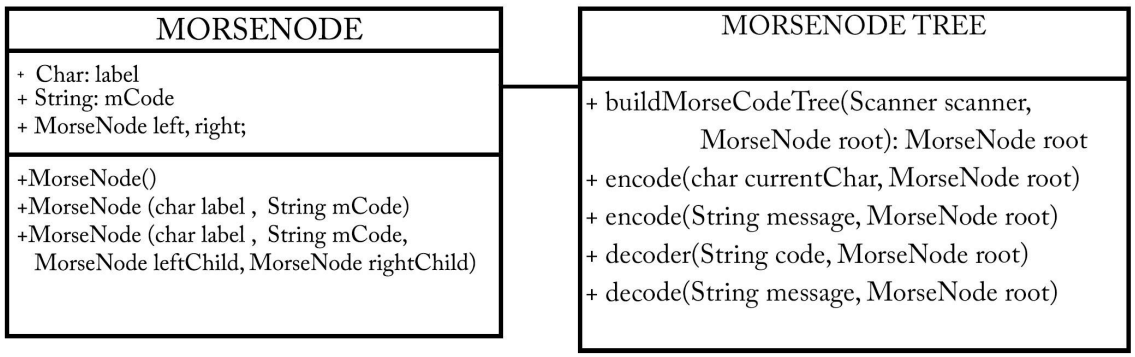
Then the user can input a message, either in English or in Morse code, and the translator will output the translation into the console for them. Then they can start over if they wish, or they can quit the program.

In the Encoder method, we take in a single character and then the root of the tree. We recursively search the whole tree for the Node with the label that matches that character, and then print out the MorseCode that is also stored within the same Node. Then, we utilize a wrapper method that changes every letter in the input message to lowercase, and encodes the whole message by calling the previous encode method with each of the characters in the message. The result is the translation of the message in MorseCode.

The decode method takes a string of morse code from user input and splits the codes into an array by a delimiter (space). For each code in the array we call the decoder method, which recursively searches the binary tree for a match, and then prints the letter stored in that node.

UML Class Diagram

The following is a UML Diagram that outlines our Morse Code Translator:



Test Cases

Test Case 1:

The first test case shows a message in English translated to Morse code. Spaces and capital letters are accepted.

Message: The quick brown fox

Expected result: - .... . --.- ..- .. -.-. -.- -... .-. --- .-- -. ..-. --- -..-

Actual result:

```
          Welcome to the Morse Code Translator
Please Choose your Menu Option by choosing a Number to enter.
 1. Enter an English message to Translate into Morse code
 2. Enter a Morse code message to Translate into English
 3. Quit
Enter here:
1
Enter Message here: The quick brown fox
Translation: - .... . --.- ..- .. -.-. -.- -... .-. --- .-- -. ..-. --- -..-
```

Conclusion: Expected result matches actual result.

Test Case 2:

The second test case shows a message in Morse code translated to English.

Message: .--- ..- -- .-. ... --- ...- . .-. - .... . .-.. .- --.. -.-- -.. --- --.

Expected result: jumpsoverthelazydog

Actual result:

```
          Welcome to the Morse Code Translator
Please Choose your Menu Option by choosing a Number to enter.
 1. Enter an English message to Translate into Morse code
 2. Enter a Morse code message to Translate into English
 3. Quit
Enter here:
2
Enter Message here:
.--- ..- -- .-. ... --- ...- . .-. - .... . .-.. .- --.. -.-- -.. --- --.
Translation: jumpsoverthelazydog
```

Conclusion: Expected result matches actual result.

Project Contributions

The contributions of each Team Member are as follows:

Rachael Mallinson

1. MorseNode Class
2. Encoder method
3. Wrapper for encode
4. UML diagram

Sofia Nikas

1. Decoder method
2. Decode method
3. Test Cases

Aleksis Martin

1. Build Morse Code Tree method
2. Ideas for Future Improvements

Ideas for Future Improvement

The following are ideas for improvements we predict we could make to our Morse Code Translator given an adequate amount of time.

1.) The user experience might be better if the menu were just to prompt them to input either a Morse code message or an English method with a special phrase or input to end the program. The program would translate it to the opposite based on what was input. This could eliminate a step, and make the process faster for the user. It would eliminate a program crash if the user didn't enter an integer at the start of the program. Also if the user inputs the wrong type of string (English text in the morse decoder (menu option 2) or morse code in the English encoder(menu option 1)) the program would not simply return an empty string.

2.) We could find a way to recursively build the Alphabet/Morse code tree. This would save on the amount of code written, and maybe processing time.

3.) Our code takes in the nextLine() of a user input, whether it's english or morse code. This allows the program to translate several words (sentences or phrases) at once without issue. It would be nice to have the translator output the morse code phrase or sentence with some delimiter to represent spaces in between the words of the encoded morse code, i.e. instead of "hello world" being translated to (.... . .-.. .-.. --- .-- --- .-. .-.. -..), use a character such as, '/', to space out the words "hello '/' world" (.... . .-.. .-.. --- / .-- --- .-. .-.. -..). This also applies to translating morse code to english. Our code returns a string but with no spaces. If you enter an encoded phrase it comes back all as one string, i.e. (morse) "quick brown fox" returns (english) "quickbrownfox".

This concludes our Project 3B Report. Thank you for this opportunity and for your attention.