**CS 303 Project 2**

*Lauren Magee and Ben Teig*

**Assumptions:**

We chose to do Project 2B, the Morse Code Option. The first time we went through the project overview we assumed that the lines between the dots of the Morse Code were dashes. However, upon further review and opening the Dropbox link, we decided that they were supposed to be underscores. For the text files uploaded, the dots need to be periods and the lines need to be underscores to be compatible with our functions. Another feature we created was the user screen even though it was stated as not being necessary on the project guidelines. This main menu allows the user to utilize both features of our program, the decoding and encoding of Morse Code. We assumed this would be the most efficient way to execute the appropriate portion of our code and allow the grader to test a variety of different submissions. Lastly, we decided to use a structure instead of a class to create our binary tree. Therefore, there are no header files included within our project. Everything else included in our program aligns with the 2B Project Guidelines.

**UML Diagram:**

**Algorithm:**

1. Set up Struct to create Binary Tree structure (left and right node plus value).
2. Make a function to create new nodes for tree (specifically string nodes).
3. Make a function to insert the new nodes into the binary tree.
4. Create a function to compare the values of the new nodes and then set either left or right from its parent node.
5. Make an Inorder Traversal function to ensure all the nodes are being inserted and situated in the correct spaces. This function will be commented out when we turn it in but uncomment it and the associated function call to check our Binary Tree Order.
6. Within the main, create and insert all Morse Code alphabet strings.
7. Create map and insert key-value pairs, with the keys being the encoded alphabet letters and the values being numbers. **EX:** morseDecoder[".\_"] = 1;
8. Create a main menu with two options for User Input with the first option as (1) Uploading a text file and having our program decode it and the second option as (2) Typing in a word and having it become encoded.
9. If the first option is selected, the user will be prompted to enter the path to the text file they wish to decode. Included in the project file is an “Example.txt.” file that can be used for this.
10. Once the path is entered, it is checked for a valid filename. If it is not valid, the path is requested again.
11. Once the file is opened, it is read into the program. The function extracts a string of characters until it encounters a whitespace. The string is then passed to the morseDecoder map to return an int value. This int value is then used to look up the corresponding letter in the alphabet by index using an array. The letter is then output to the console.
12. Step 10 is completed until the end of file.
13. If the second option is selected, the user will be prompted to enter a string. The string will then be broken up character by character and put into a character array.
14. Then, the character array will be compared to an alphabet array to find the index value of their mutual letter.
15. Using the index value, the Morse code equivalent of that letter will be found and outputted back to the user.
16. Repeat Steps 13-14 until the whole string has been encoded.
17. End of Program.

**Efficiency of Algorithms:**

The applicable Big O efficiencies of our major functions are commented right beside them within our program. We believe there is always a way to make a program more efficient in terms of space and time complexity. However, with our current range of skills and abilities we believe we constructed our program in the most efficient way possible.

**References:**

*Bing*, Microsoft, 23 Feb. 2014, <https://www.bing.com/videos/search?q=how+to+code+a+binary+tree+c++&&view=detail&mid=6BD05489F8FF691622F96BD05489F8FF691622F9&FORM=VRDGAR>.

*Bing*, Microsoft, 9 Mar. 2014, <https://www.bing.com/videos/search?q=how+to+code+a+binary+tree&&view=detail&mid=B4DE8FC113717E93509EB4DE8FC113717E93509E&FORM=VRDGAR>.