Team 1 Project 3B

### PREPARED BY

Joe Simon

Eric Vaughn

Jordan Pham

2) System Design

Our system has two .cpp files—Main.cpp and Morse\_Code.cpp—and one header file, Morse\_Code.h. The Main file contains the int main() function which creates a Morse\_Code object first. The user is then asked to enter a series of letters to be encoded and decoded by Morse code. These letters are stored within the string set\_of\_letters. The encoded\_message string is then initialized by the firsts’ encode function of the series of letters. The original message and the encoded\_message are then output to the user. The encoded\_message is then decoded under firsts’ decode function, which is also output to the user.

After the Morse\_Code object is created in the main function, the create\_binary\_tree() function is called within the default constructor. This function reads from the Morse\_Code.txt file, which contains the Morse codes for each lowercase letter. Within a while loop, it calls the create\_node() function, which adds a node, containing a letter from the .txt file, to the binary\_tree\_root. Once the encode function is called, it goes through each letter within the parameter string letters and passes the letter into the find\_code\_for\_specific\_letter() function. This function recursively finds the letter and then assigns the string temp\_c variable with the code. In the encode function, the temp\_c variable is added to a string of codes, which is then returned. When the decode function is called, it also recursively finds the letter in the binary tree, adds the letter to a string of letters, and then returns those letters.

There are several data structures used in our system. First, we used the string data structure for storing several variables in the Main program and the Morse\_Code files such as letters, codes, decoded\_message, etc. Second, we used the iterator data structure for looping through strings. Third, we used a binary tree to create the Morse code tree with the node binary\_tree\_root as its root from the text file Morse\_Code.txt.

3) UML Class Diagram

4) Two Test Cases

# Test Case 1:

* Expected output when encoded:
  + .- -... -.-. -.. . ..-. --. .... .. .--- -.- .-.. -- -. --- .--. --.- .-. ... - ..- ...- .-- -..- -.-- --..
* Actual output when encoded:
  + .- -... -.-. -.. . ..-. --. .... .. .--- -.- .-.. -- -. --- .--. --.- .-. ... - ..- ...- .-- -..- -.-- --..
* Expected output when decoded:
  + abcdefghijklmnopqrstuvwxyz
* Actual output when decoded:
  + abcdefghijklmnopqrstuvwxyz

A screen shot of a computer

Description automatically generated

# Test Case 2:

* Expected output when encoded:
  + . .. ... .... ...- ..- ..-. .- .-. .-.. .-- .--. .--- - -. -.. -... -..- -.- -.-. -.-- -- --. --.. --.- ---
* Actual output when encoded:
  + . .. ... .... ...- ..- ..-. .- .-. .-.. .-- .--. .--- - -. -.. -... -..- -.- -.-. -.-- -- --. --.. --.- ---
* Expected output when decoded:
  + eishvufarlwpjtndbxkcymgzqo
* Actual output when decoded:
  + eishvufarlwpjtndbxkcymgzqo

A screenshot of a computer

Description automatically generated

For both the encoded function and the decoded function, the expected and actual outputs are the same (besides the formatting error in test case 2, though this is a Microsoft Word problem and not a code problem).

5) Contributions

1. Joe
2. Eric
3. Jordan

6) Improvements

1. Include functionality for more symbols and capital letters
2. More team project meetings
3. More comments inside of each function
4. Improve the efficiency of the `encode` function
5. Make the main function more user-friendly (GUI interface)