Documentation

Dear reader,

This is the Documentation File for my C Project that encodes and decodes Morse Code using file reading and binary tree. Also, the project fulfills the requirements of CPROG Portal, such as command line control. The traversal through the binary tree is done through structures created by me, named “Node”.

This Project relies on File Handling, Binary Trees, and Dynamic Memory Allocation.

The 12 functions used in this Project are described below.

Functions

This Project uses 12 functions:

1. **void showIntro()**

   This function is used to print a simple Hello and Welcome Message to the User.

2. **int showTypeOfReadingMenu();**

   This function is used to print the first menu to the User, and they must choose if they want to enter English Text or Morse Code by themselves, or to read it from a file. The User enter either 1 (to read from file), or 2 (to enter text/code themselves). The number that the User enters is returned by the function to determine latter the type of reading.

3. **int showHandlerMenu();**

   This functios is used to print the second menu to the User, and they must choose if they want to encode English Text to Morse Code – option 1, decode Morse Code into English Text – option 2, or print statistics about the Morse Code (number of dashes and hyphens) – option 3. The number that the User enters is returned by the function to determine latter the type of reading.

4. **Node* fillTree();**

   This function is used to initialize of binary tree and fill it with English Letters. First the function creates the root of the tree of type Node. Then allocates memory for the root all the nodes in the subtrees of root. After this is done, nearly all characters of each node is assigned an English Letter, according to international agreement. The function returns the reference to the root node, using pointers.

5. **void morseIntoTextFileReading(Node *root);**

   This function decodes Morse Code into English Text. It takes the root of the tree as argument and uses another helping structure (temp) to iterate through the tree, so as not to lose reference of the root. For every dot, temp refers to its left child, and for every hyphen, temp refers to its right child. When any other character is encountered (probably a space), the string value of the temp is printed and temp points again to the root to start calculating the next English Letter.

6. **void morseIntoTextCommandPrompt(Node *root);**

   This function is very similar to function nr 5. The Morse Code is input from the user and the same logic as above is used to decode it into English Text.

7. **void charToMorse(char ch);**

   This function is used to "translate" an English Alphabet Letter into Morse Code. The switch-case control statement is used, and for any value that the parameter ch has, the function prints its Morse Code equivalent.

8. **void textIntoMorseFileReading();**

   This function is used to encode English Text into Morse Code. The English Text is read from a file and every character in the file is passed as an argument in function nr 7, to print its Morse Code equivalent.

9. **void textIntoMorseCommandPrompt();**

   This function does the same thing as function nr 8, but the text is not read from any file. Instead, the user enters the text themselves and the String is iterated char by char and the Morse Code of each char is printed.

10. **void statisticsFileReading();**

    This function gives statistics about a Morse Code written in a file. The function uses 2 counters and checks each char in the file. If the char is a dot, the counter for the dots is incremented by one. If the char is the hyphen, the counter for the hyphens is incremented by one. In the end, values of these counters are printed to the screen.

11. **void statisticsCommandPrompt();**

    This function does the same thing as functions nr 10. The only difference is that the Morse Code is entered by the User as input.

12. **void run();**

    This function is the only function called in main() function. It is used to print welcome message to the user, print menus, get any information from the user, call any function needed, and print results to the screen. Command line switch control is done through this function.

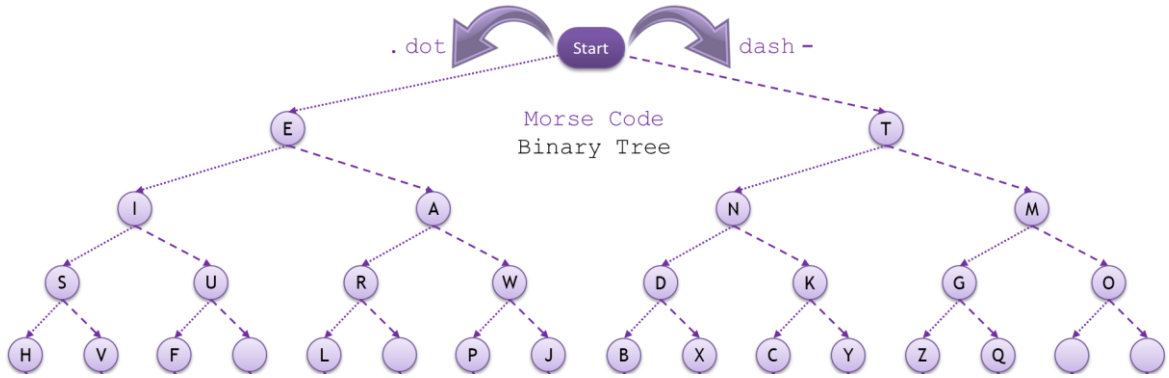
Read from file

Two files are created in the Project named "inputText.txt", used to write English Text, and "inputMorse.txt", used to write Morse Code. The used can edit these files if he wants to use the Program via File Handling method.

Command Prompt

When the Program starts, the user is first introduced and welcomed to the Program. Then he/she is given the possibility to choose which reading method he/she wants to use. Then he/she is given possibility to choose if he/she want to Encode, Decode, or see statistics.

Binary Tree



As shown in the picture, Morse Code is well handled via Binary Trees. The logic after the tree is that, whenever a dot is encountered, we move to the left child of the node that we currently are, and whenever a hyphen is encountered, we move to the right child of the node that we currently are. When we finish with the Morse Code of the current English Alphabet Letter, the char in the node that we are in the corresponding English Letter of the Morse Code.

Command Line Switch

Switch-case / If / If..else statements are used to determine which path of the program to be executed, according to the current condition of the program, or to user's input. If first condition is true, then that part of code is executed, and the rest of the code inside the control statement is neglected. If not, the second condition is checked, and if true, then that part of code is executed, and the rest of the code inside the control statement is neglected. This logic continues until the end of the control statement block of code.