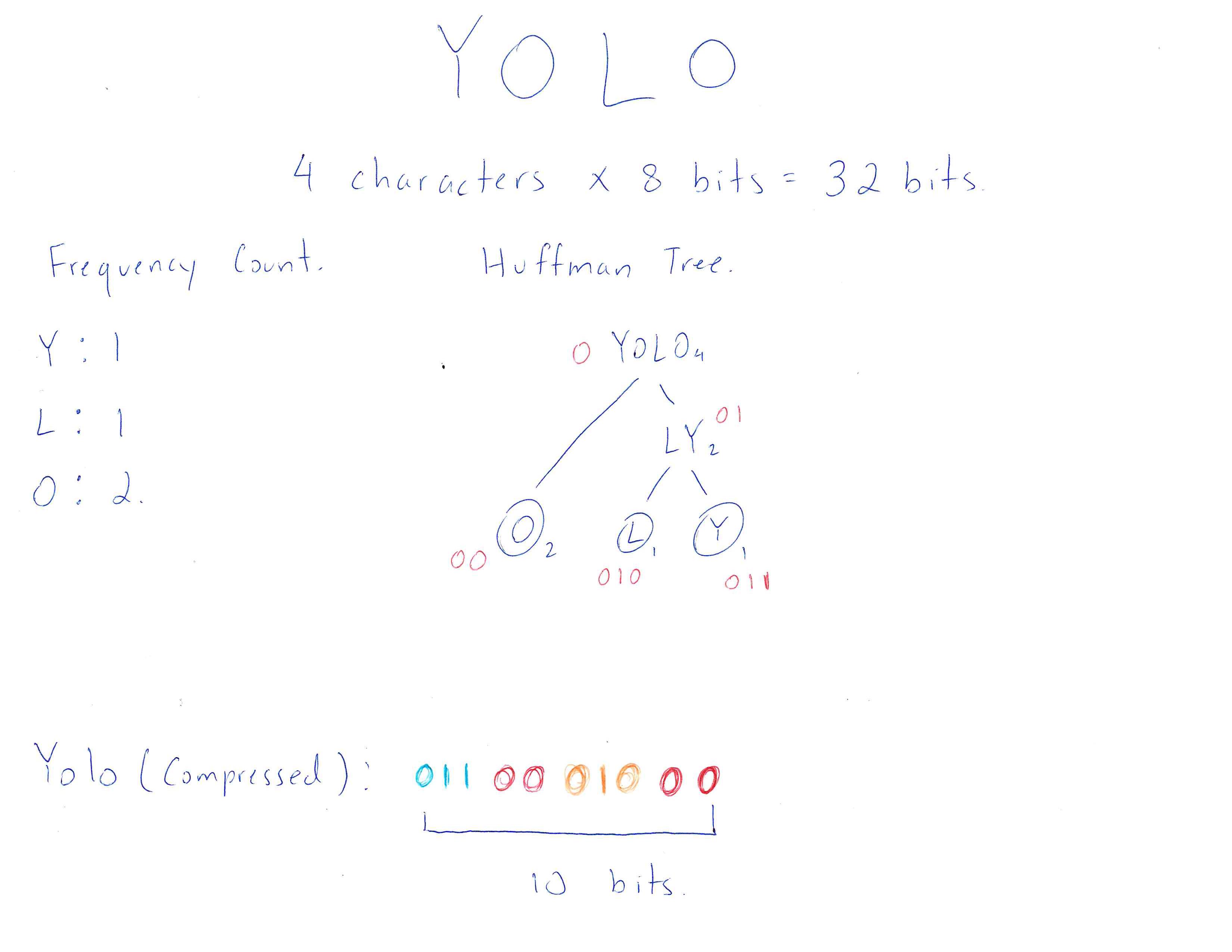
Huffman Coder Documentation

This is a graphic from our presentation to explain Huffman encoding.

Essentially, we compressed our beta into 1 file with 2 classes. We ask for user input, which is saved as a string. We first create an array list of characters that stores all the unique characters within the phrase. Next we found the counts of each letter within the phrase. With this information we created a HuffmanList, a linked list of nodes that contain the character and the frequency. Following, we created the HuffmanTree, which we built by removing and merging the nodes with the two lowest frequencies, and this new node is the parent. We add the parent node to the linked list. We recursively call the HuffmanTree method and this builds the tree. Finally, we binary encoded all the characters by encoding a 0 for each left branch and a 1 for each right branch, and print the final encoded string. The console looks as such:

To explain why the binary encoding is different from the example, it is primarily due to the order of “L” and “Y”. The computer puts “L” as the end node instead of “Y”, which is what we did in the demonstration. Additionally, in the example, we added an extra “0” in front of all the codes.

