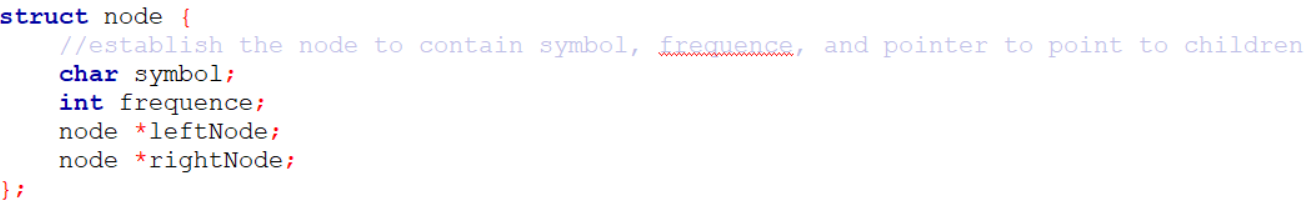
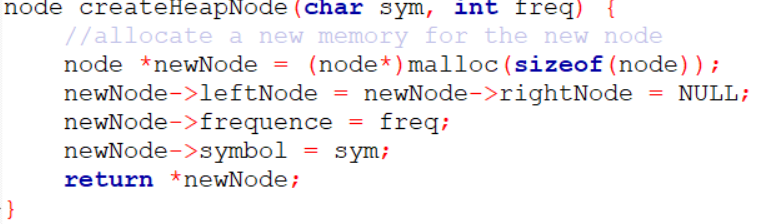
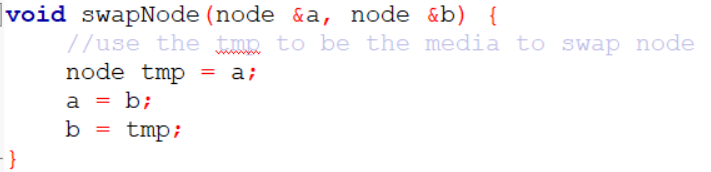
Use the struct to store each information of the node.



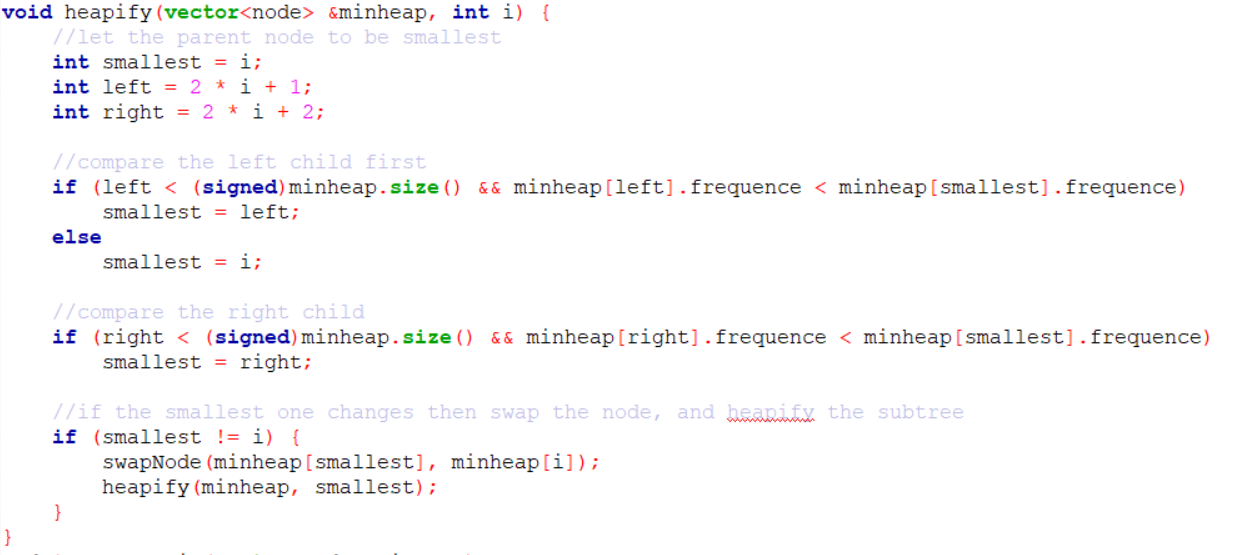
Use “createHeapNode” function to create a new node with the given information.



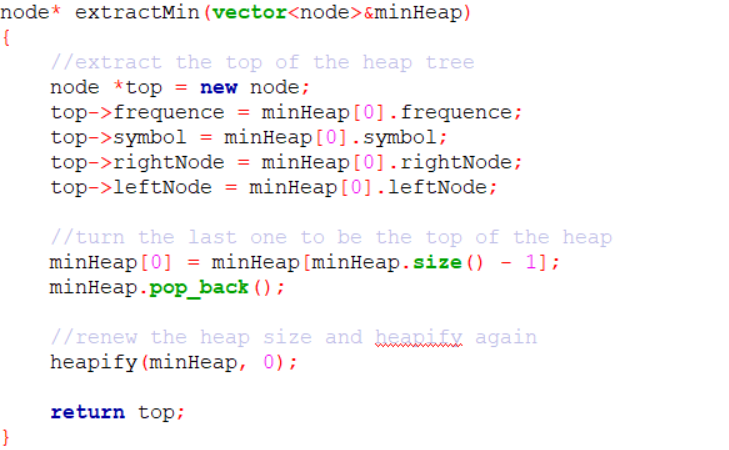
Use “sawpNode” to exchange two nodes.



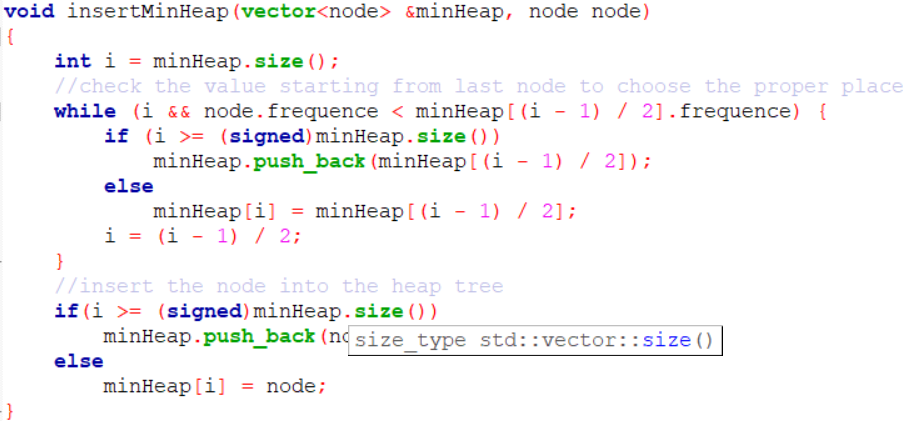
Use “”heapify” function to implement the minheap tree rule.



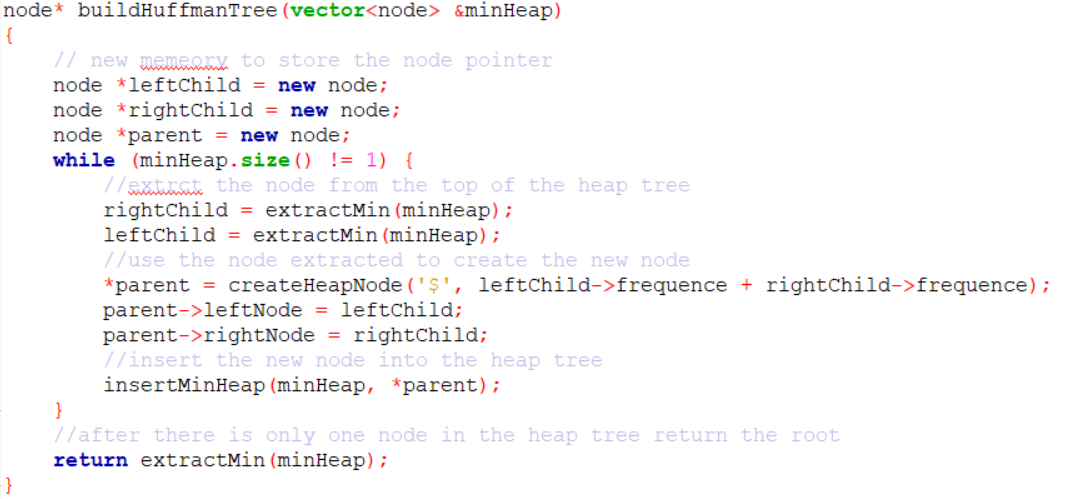
Extrac the top node from the heap and return the node to the place it calls.



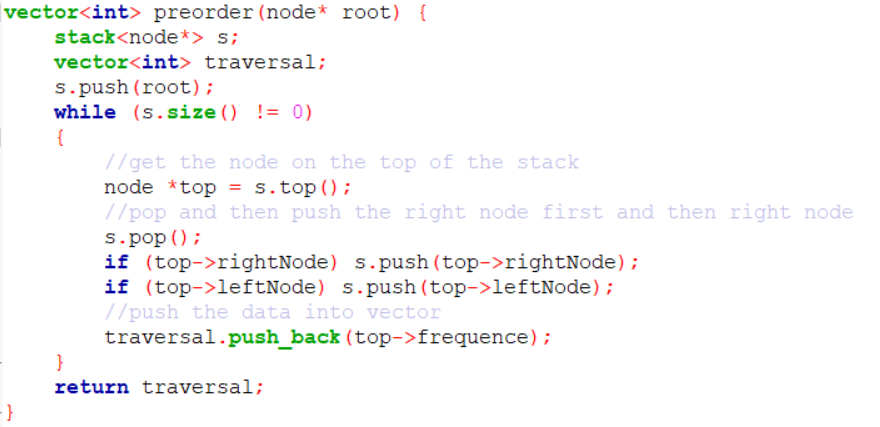
Use this function below to insert a new node into the heap tree



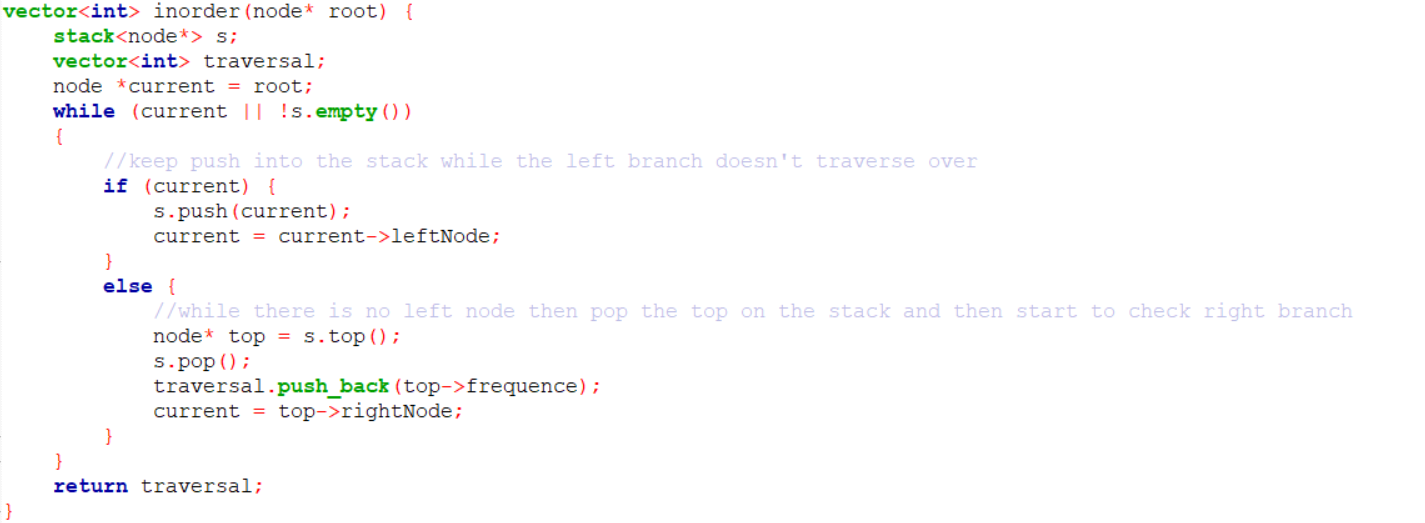
Use the function below to create the Huffman tree which follows the rule to create a new node by adding the frequency of two top node extracted from the heap tree.



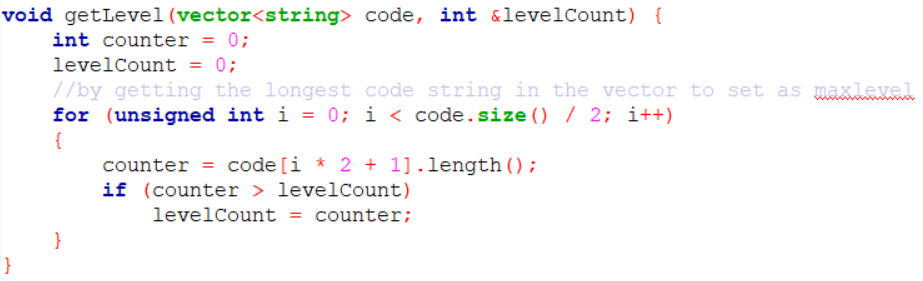
Use the function below to return a int vector which store the results after preorder traversal



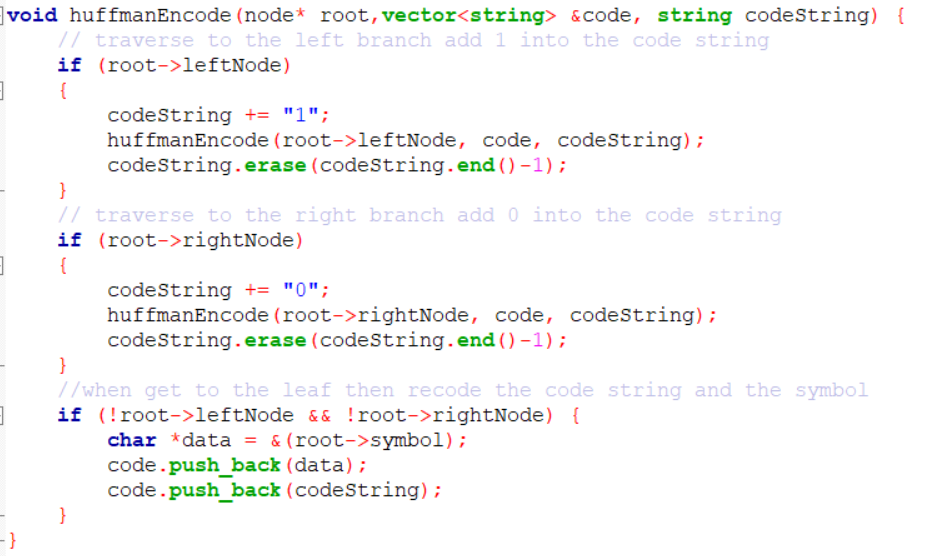
Use the function below to do the same thing as above but change the traversal way to inorder traversal



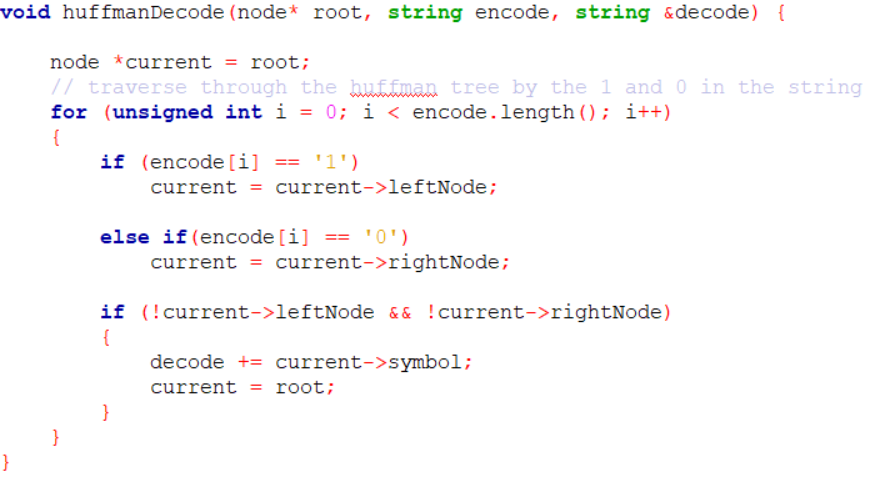
Use the function to get the max level of the Huffman tree by get the longest string of the Huffman code.



According to the rule that when traverse to the left branch then add “1”, the right branch then add “0”, build the code string of each symbol. And since the recursive method is used, the last char must be removed after the string returned.

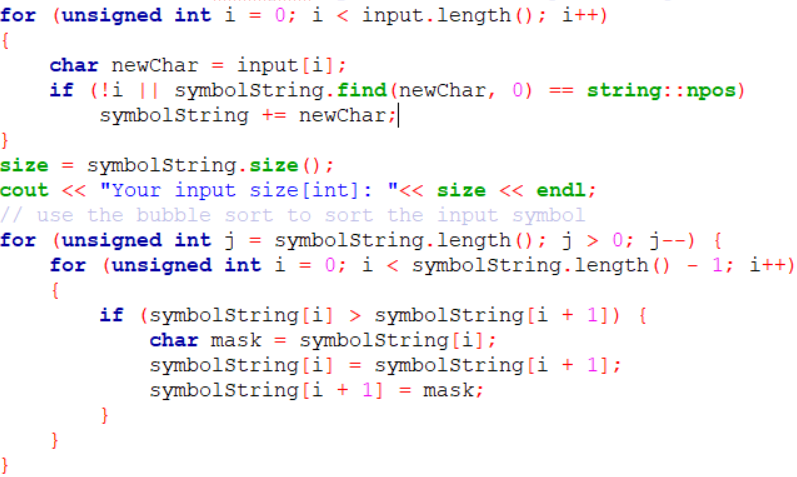


Use the function below to decode the string input by traverse to left if the input char equals to “1” and to right if to “0”, and add the symbol got from the traversal to the decode string.

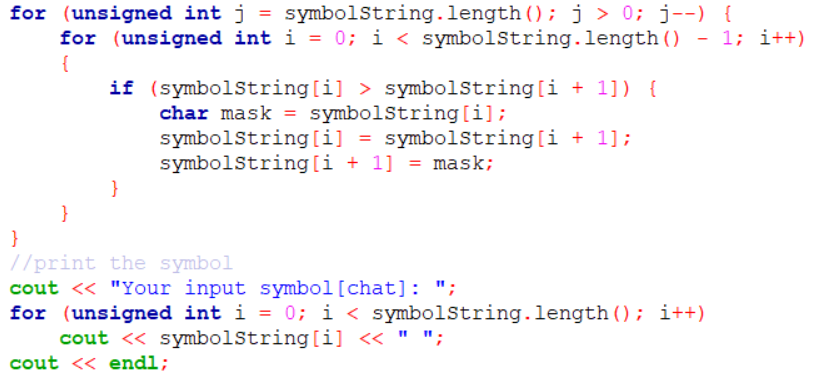


Preprocess the data input:

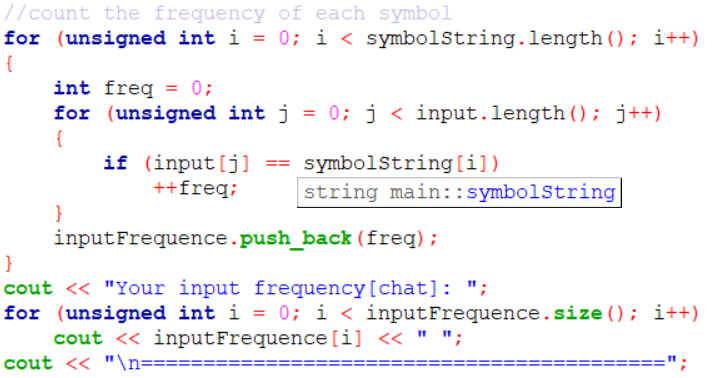
Get different chars and store in the “symbolString”



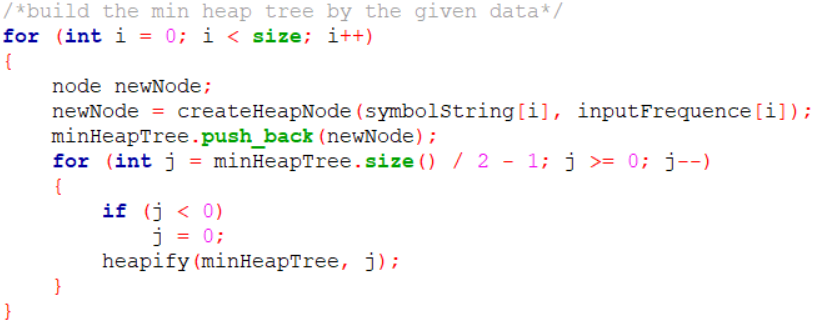
Reorder the symbol and print in proper order



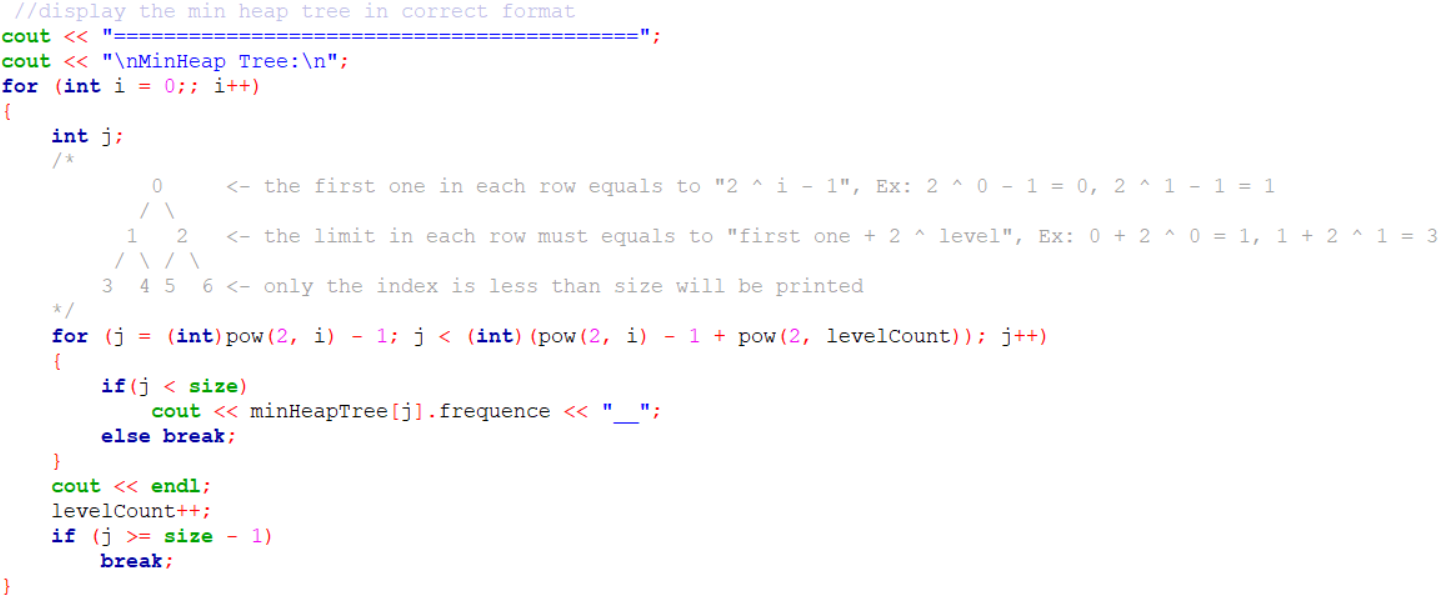
Count the frequency of each symbol and then print out



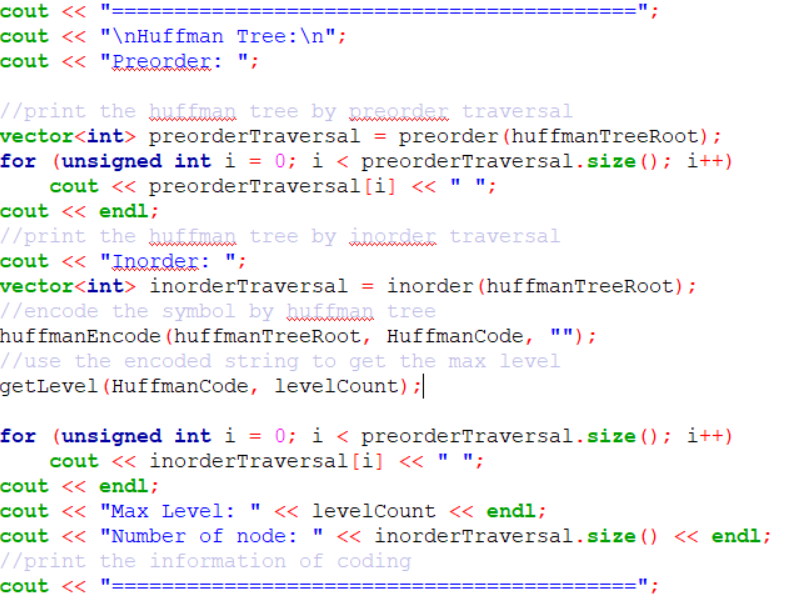
Build the min heap tree by creating new nodes and putting the nodes into the function to heapify.



Print the minheap tree by level



Build the Huffman tree and print out the tree by inorder and preorder traversal only pass the parameter of the root pointer, and create the encoded string to put into the function to get the max level of the tree.



Get the decoded string by pass the encoded string as parameter, and print the string out.

