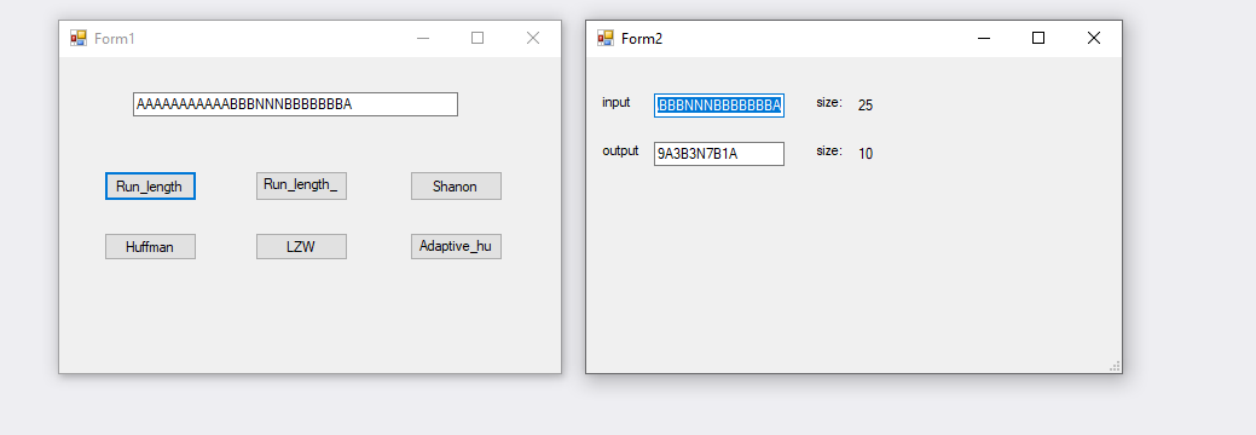
Engy Makram Malak

Haidy Thabet Roushdy

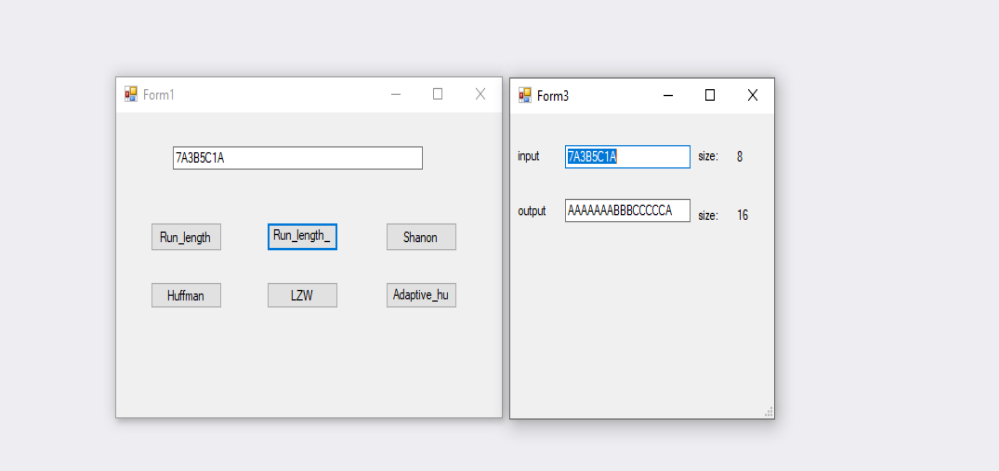
Run Length Encoding: In run length encoding, the input string is encoded by replacing a substring of repeated character in the string by the character followed by its count. If the character is single and is non-repeating than it's count is not added. For Example, if the input string is "wwwwaaadexxxxxx", then the function should return "w4a3dex6"

Examples:



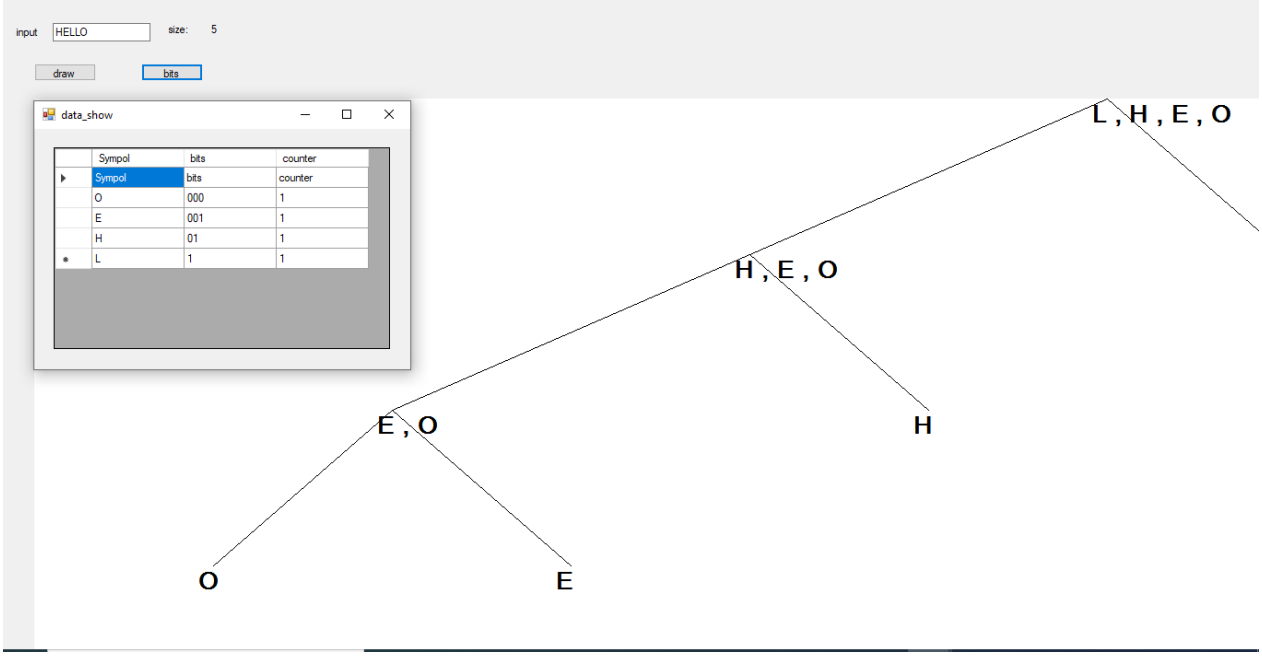
Decoding an RLE-encoded stream of data is actually even easier than encoding it. Like before, you iterate through the data stream one character at a time. If you see a numeric character then you add it to your `count`, and if you see a non-numeric character then you add `count` of those characters to your decoding, which is returned to the caller once you iterate through all of the input`data`.

Examples:



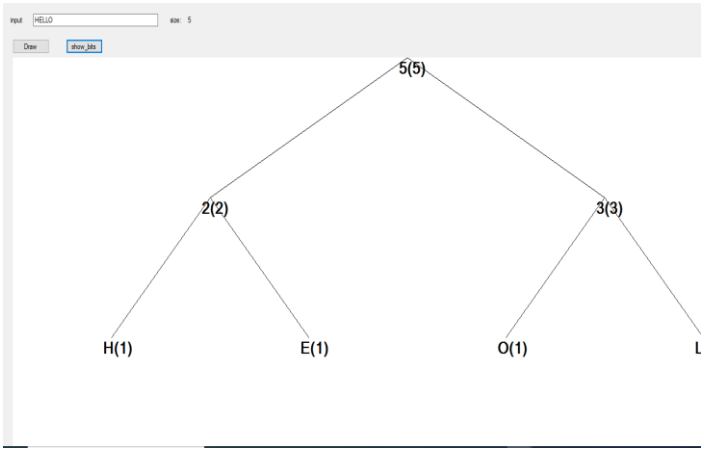
Shannon–Fano codes: are suboptimal in the sense that they do not always achieve the lowest possible expected codeword length, as Huffman coding does.[1] However, Shannon–Fano codes have an expected codeword length within 1 bit of optimal. Fano's method usually produces encoding with shorter expected lengths than Shannon's method. However, Shannon's method is easier to analyse theoretically.

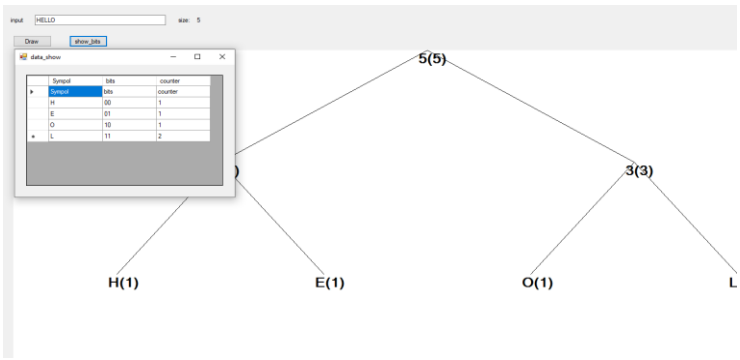
Examples:



Huffman coding ：is a lossless data compression algorithm. The idea is to assign variable-length codes to input characters, lengths of the assigned codes are based on the frequencies of corresponding characters. The most frequent character gets the smallest code and the least frequent character gets the largest code.

Examples:

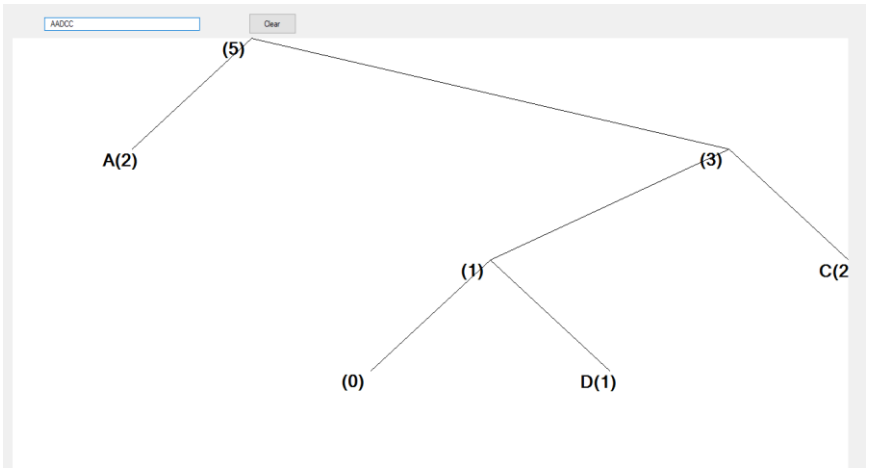




Adaptive Huffman coding (also called **Dynamic Huffman coding**) is an adaptive coding technique based on Huffman coding. It permits building the code as the symbols are being transmitted, having no initial knowledge of source distribution, that allows one-pass encoding and adaptation to changing conditions in data.

The benefit of one-pass procedure is that the source can be encoded in real time, though it becomes more sensitive to transmission errors, since just a single loss ruins the whole code.

Examples:



The lzw compression algorithm is the following: as the input data is being processed, a dictionary keeps a correspondence between the longest encountered words and a list of code values. The words are replaced by their corresponding codes and so the input file is compressed. Therefore, the efficiency of the algorithm increases as the number of long, repetitive words in the input data increases.

Examples: