Huffman Encoding Design Document

**Methodology and algorithm.** To encode the input the stream the contents of the input file are looped through one character and a time. The character is added to a new Node instance and placed into a List of Nodes. Then the List is sorted, and then looped through to determine which Nodes can be combined into Left and Right leaf nodes. If the Node can be combined, the nodes are recursively added to a Tree data structure until there is a single node remaining. The list of nodes is then recursively walked through, setting each leaf node to a Boolean value based on if the node is a Left or Right leaf node. The Boolean values are added to a BitArray and returned from the encoding process.

To decode the values, the BitArray from the encoding process is looped through recursively checking each bit against the stored root nodes child nodes. If the bit is on and the current nodes right node is set, then the current node becomes the current nodes right node. Conversely if the bit is off and the current nodes left node is set then the current node becomes the current nodes left node. Finally, if the current node is a leaf node the current nodes value is added to the decoded string.

**Data structures used.** The program makes use of the List, Dictionary, and BitArray data structures. Lists are used to hold the nodes during the encoding process as well as for temporary holders during the process in which the ordered nodes are combined into left and right leaf nodes. A Dictionary is used for quick look up during the first part of the Node Tree hydration process to determine the counts of each character.

**Testing Plan.** The full wap document can take a while to get through so smaller wap1.txt file is included with just the first line of text. This can be used to quickly see if the program is functioning correctly. Of course, any ASCII encoded file can be used to test the encoder. To test the output from the encoding process and the output from the stream the two strings are compared using String.Equals(). The results are printed to the screen so the user can quickly assess whether the results are correct.

**How to use.** Run the executable from the command line. The program will ask the user to enter in the name of a file. Depending on the system the runtime location may be different so if the program cannot find the file it will output the exception message which will have the location of the file it is looking for. The program will output diagnostic information along the way to give the user an idea of where it is during program execution. Time diffs are also provided to give the user some feedback on how quickly the encoder and decoder works.