# CS 211 PA #1

In this task, you will implement a program that uses Huffman codes (book CH 16.3) to compress files. I will cover Huffman encoding on the day that this assignment is handed out. However, to get a complete sense of how Huffman codes work, ***I strongly suggest that you read the book***. For this assignment, I provide quite a bit of starter code. Furthermore, included in this document is a working executable of my solution for you to test against. To complete this assignment, you must use the following data structures:

* STL vector for bit and text storage
* STL Priority Queue for Huffman forest storage
* STL unordered\_map for­­ storage of Huffman Maps
* Custom HuffmanNode, HuffmanInternalNode, HuffmanLeafNode, and HuffmanTree classes to represent a Huffman Tree.

# Required Functionality

You are required to implement the following functionality in PA1.cpp.

### huffmanTreeFromText

Builds a Huffman Tree from the supplied vector of strings. This function implement's Huffman's Algorithm as specified in page 431 of the book. **IMPORTANT!** In order for your tree to be the same as mine, when merging trees, you must make sure to place the smallest tree on the left side! Example:

|  |
| --- |
| Input string:  AAABBC |
| Resulting tree: |

### huffmanTreeFromMap

Generates a Huffman Tree based on the supplied Huffman Map. Recall that a Huffman Map contains a series of codes (e.g. 'a' => 001). Each digit (0,1) in a given code corresponds to a left branch for 0 and right branch for 1. I used recursion to solve this problem. Example:

|  |
| --- |
| Input map:  A0  B11  C01 |
| Resulting tree: |

### huffmanEncodingMapFromTree

Generates a Huffman Map based on the supplied Huffman Tree. Again, recall that a Huffman Map contains a series of codes (e.g. 'a' => 001). Each digit (0,1) in a given code corresponds to a left branch for 0 and right branch for 1. As such, a given code represents a pre-order traversal of that bit of the tree. I used recursion to solve this problem. Example:

|  |
| --- |
| Input tree |
| Encoding map:  A0  B11  C01 |

### writeEncodingMapToFile

Writes the supplied encoding map to a file. My map file has one association per line (e.g. 'a' and 001). Note that the first character in each line is the original character. All subsequent characters represent the character's encoded value.

### readEncodingMapFromFile

Creates a Huffman Map from the supplied file. Essentially, this is the inverse of writeEncodingMapToFile. Be sure to use my StringSplitter class to make your life easier!

### decodeBits

Uses the supplied vector of bits (bools) and Huffman Map to convert the vector of bits back into text. To solve this problem, I converted the Huffman Map into a Huffman Tree and used tree traversals to convert the bits back into text. Example:

|  |
| --- |
| Input bits:  000111101  Input tree: |
| Result:  AAABBCCC |

### toBinary

Uses the supplied Huffman Map to convert the supplied text into a vector of bits (bools). Example:

|  |
| --- |
| Input map:  A0  B11  C01  Input string:  AAABBC |
| Result:  000111101 |

## Header Comment, and Formatting

1. Be sure to modify the file header comment at the top of your program to indicate your name, student ID, completion time, and the names of any individuals that you collaborated with on the assignment.
2. Remember to follow the basic coding style guide. For a list of basic rules, [see my website](http://adamcarter.com/teaching/cpts121/style) or examine my example files from previous assignments and labs.

## Reflection Essay

In addition to the programming tasks listed above, your submission must include an essay that reflects on your experiences with this homework. This essay must be at least 350 words long. Note that the focus of this paper should be on your reflection, ***not*** on structure (e.g. introductory paragraph, conclusion, etc.). The essay is graded on content (i.e. it shows deep though) rather than syntax (e.g. spelling) and structure. Below are some prompts that can be used to get you thinking. Feel free to use these or to make up your own.

* Describe a particular struggle that you overcame when working on this programming assignment.
* Conversely, describe an issue with your assignment that you were unable to resolve.
* Provide advice to a future student on how he or she might succeed on this assignment.
* Describe the most fun aspect of the assignment.
* Describe the most challenging aspect of the assignment.
* Describe the most difficult aspect of the assignment to understand.
* Provide any suggestions for improving the assignment in the future.

## Deliverables

The final version of your program must be uploaded through Canvas no later than midnight on Wednesday, February 13, 2019.

## Grading Criteria

Your assignment will be judged by the following criteria:

### Test Cases (80pts)

* Your program successfully passes all test cases

### Reflection essay (10pts)

* Your reflection meets the minimum requirements as specified earlier in this document.

### Runtime complexity analysis (10pts)

* You correctly identify the runtime complexity of your functions.