Project Report

Algorithms Used:
Huffing Program:

The basic structure of the program is that the main function essentially calls the encode function which leads to everything else being executed. Following are concise descriptions of the important functions used in this program:
1. The encode list function takes the input filename and the output filename as inputs after which it first calculates the frequency of each character it reads each character from the input file. The structures that are used are ListNode, TreeNode and ASCII_Freq. These are essential in making the codebook which will be used to encode. After the codebook has been built I used bit packing operations to compress them and then write them to the output file.

2. The compress function uses bitwise operations and maps them all together. Wbit is used to keep track of the specific bit you're on and cbyte is used to keep track of the current byte as you cycle through.

3. The wBit function is used to write bit by bit to the output file. It starts packing from 0 (left most) to 7(right most). Once it packs a byte it writes the byte to the output file.

4. The count_Freq is used to calculate the weights of each ASCII character. It reads it and then simply has a counter which keeps track of the number of times it appears.

5. sort_freq uses qsort function to sort all the weights in order. It calls a helper function which typecasts and returns the difference between the two. This is an argument required for qsort.

6. merge_TreeNode is used to merge two different tree nodes. It merges them in post order, with left, right and then zero at the top.

7. build_list is used to actually start building the list. It first finds the first index with a non-zero frequency and then it creates a linked list where each node will point to a tree node. It does this while making sure if a letter is present or not by using while (ind< 128).

8. theight is used to get the height of the tree, it traverses through the tree until it reaches 0 after which it returns the height. It calls the theight_help function which basically does it.

9. tleaf is used to calculate the number of leaf nodes, it traverses until it finds a leaf node and then increment a counter to keep track of it. It also calls the tleaf_help function which does this by traversing through the tree until it finds a leaf node.

10. build_cbook is used to build the codebook that we will use to actually do huffman compression's. It traverses through the tree recursively until it finds a leaf node. When a leaf node is encountered it stores the character. It populates the codebook by columns and rows.

Unhuff Program:
For this most of the functions that I used were the same as the ones used in the huffing program. Again, the main file calls the decode function and rebuilds the tree. It then uses the code book to get the original characters. This part of my code is not working properly.

Results:

File #	File Size	Compressed Size
1	14 bytes	6 bytes
2	77 bytes	22 bytes
3	1016 bytes	256 bytes
4	13.1 kb	3.9 kb
5	55.0 kb	17.6 kb

Notes:
Learned huffman programming in ECE264 and also used Intermediate C Programming book by Prof Yung-Hsiang Lu to work on both huff.c and unhuff.c.