ECE368

PROJECT 2

GUANSHI HE

1. The format of your compressed file

The format of my compressed file would be

[Header]

I. The secret code “HUFF” for checking if the file is the very file that being zipped by my huff.c

II. The number of Huffman table node count

III. The bite count of the source file

IV. Huffman table of all the nodes

V. Huffman table data

2. Whether and how a Huffman tree is used in both your huff and unhuff programs.

Yes, I used Huffman tree in my programs. By counting the frequency of each character in the file, then put then into the preset huff tree stack and build the tree according to the frequency order. For the unhuff program, read the header bytes first and then the Huffman table data, first check if the input file is with the secret code or not, and read the basic information of the Huffman table, then traverse the tree in binary one digit at a time (if you got a ‘0’ , go to the left child of the node, and if you read a ‘1’, go to the right child). When you reach a leaf that with no child, it is the character. Repeat the previous step until the end of the file.

3. Whether you have adopted code from ECE264 or other sources. If so, in what way?

No, I have not adopted any code from ECE264.

Solution Description

Huff.c

1. Read the input file and count the frequency of each character

2. Build the binary tree according to the frequency table

3. Create an encoding map and output it to the HUFF file by putting the header file first

Unhuff.c

1. Read the header part of the file to check the secret code and the table count

2. Read the binary character one digit at a time and follow the Huffman tree to reach each character

3. Repeat step 2 until the end of the file

Results

Small size file	3.4KB
Size after Huffman coding	1.5KB
Size after Unhuff coding	3.4KB
	Compression ratio = 44.11%
Medium size file	2.5MB
Size after Huffman coding	1.5MB
Size after Unhuff coding	2.5MB
	Compression ratio = 60%
Large size file	27.8MB
Size after Huffman coding	16.2MB
Size after Unhuff coding	27.8MB
	Compression ratio = 58.27%

The compression ratio will vary with the content and size of the file. File contain many characters with different frequencies will increase the complexity of the huffman binary tree and also the size of the file after huffman coding.