Perseus Cabal (solo)

5/7/16

Comp 271 Final Project

Huffman Encoding

Project Description

This project involves implementing the Huffman Encoding algorithm. The coding is meant to take an input from the user and compress the input into binary. The code takes every character from the user input and counts the frequency of each character. All of the characters are put into a binary tree and assigned a path that encodes for its binary counterpart. The characters that have the highest frequencies are higher in the tree, and therefore require less memory (smaller binary encoding) and are easier and faster to access compared to characters that have lower frequencies. Characters that have lower frequencies are further down the tree and are encoded for by a longer binary number.

Data Structures Used

For this project, I used a struct to create a Binary Tree because it provided an easy way to create a node. Each node was assigned a character, the character’s frequency, and a left and right child, and all of these components were easily assigned with a struct. I also used a priority queue (via array). The priority queue was crucial because it allowed easy creation of the binary tree. The characters in the queue were given priority based on the frequency, with the lowest frequency getting the highest priority. This enabled easy creation of the tree (connected the nodes by links)by putting the low frequency characters at the bottom, creating a node with the sum of the frequencies, and repeating the process until all of the characters were in the tree (one node remaining). A binary tree was made and traced in order to create the encoding for each character.

How to Use the Code

The code only requires user input. When the code runs/builds, the console will ask the user for an input greater than 1 character long, if the user inputs a string less than 2 characters, the console will keep asking the user for input until the input is acceptable. When the input is longer than 1 character, the console will output the sentence that was inputted, and then output the Huffman-encoded version.

IDE and OS Used

This project was created using CodeBlocks on a Mac.