DATA STRUCTURES PROJECT REPORT:

DATA COMPRESSION

Group Member: 19k-1463 (JAHANZAIB ALI)

Section - C

Instructor: Miss Farah

Algorithm: Huffman Algorithm/Tree

Algorithm:

As it known as Huffman tree, it works look like a tree but it work like a suffix tree not fully suffix but little bit similar to suffix tree. It fulfills the need of compressing the message. Moreover, if we have a data in a file with some lines. Suppose the original size of that data is 160 bits but if we apply Huffman technique to tackle with size problem of message. The Huffman algorithm convert it’s size into approximately 115 bits. So, we have know much smaller size of message with same content. In addition, the main feature or part of Huffman algorithm is it gives less no of bits to those who came most no of times in data and assign large no of bits according to the no of characters it has. Example, Usama is a data in a file. We place them in series in a priority queue with min heap technique because as we know that less frequent data will came first than the most ones. So, U = 1 , s = 1 , m = 1 , a = 2. These numbers are the frequencies of every distinct characters in a data. Now according to the second rule of Huffman we create a Huffman tree which manage the encoding as well as decoding. Further more, we join the two least frequent leafs to make a node which is contains the joint frequency of both two characters. Like, U and s will join to create a node which will not have a name or character, all the characters of data file will only be leaf nodes.so, now U+s->node = 2 , m = 1 , a= 2. Now we have a choice we can join either m and a or U+s and m. suppose we join a and m then it will-> m+a = 2. Then the tree will like U+s->node = 2 and m+a -> node = 2, now we join U+s and m+a nodes which will be (U+s+m+a) - > node = 4; after creating this tree we assign 1 and 0 to the tree to give unique encoding version to every distinct characters. We assign 0 to left side and 1 to right side of every node till we reach leaf nodes.so, these things are for only encoding. Now we decode that encoded version with the help of that Huffman tree. Suppose U contain 00 , s = 01 upto so on. we check the encoded version which will be save on a string type variable the first character will be 0 so, we check tree then move left because all lefts have assigned with 0 and rights with 1. Then we check that the leafs come or not if it did not came we go to left side of tree and check another bit means now 0 then the encoded version is 00 which is for U as we assigned it in tree at the time of encoding. Now if it matches we write that character in a file or on console as well. So, it will works until we check all the bits of encoded version of data of file.

This is the link of the algorithm’s video that I had learned from.

**Link:** **https://www.youtube.com/watch?v=co4\_ahEDCho&t=898s**

HUFFMAN TREE OF ABOVE EXAMPLE:

frequencies

5

1

0

3

2

1

0

0

1

1

2

1

1

a

m

S

U

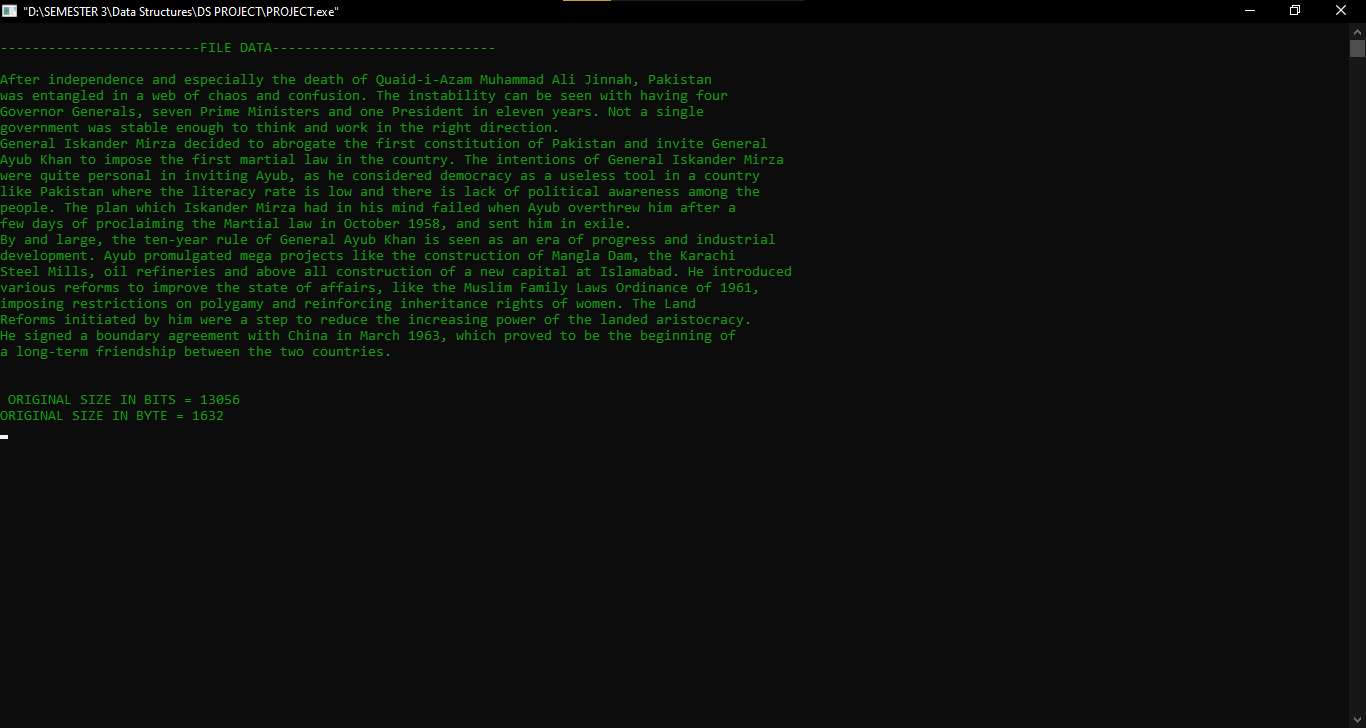
Project Overview:

I first print the introduction of myself and project and it’s basic and mostly used things in this algorithm. After this it reads data from input file and print that data first in console in front of user to show that this is the data with that given size then we apply Huffman algorithm and prints it’ same data in front of the user with less no of bits as well as bytes. After applying algorithm it prints the data in encoded version as well as decode version to make you satisfy that it is working on data file. So, after this the program will end. Thank You.

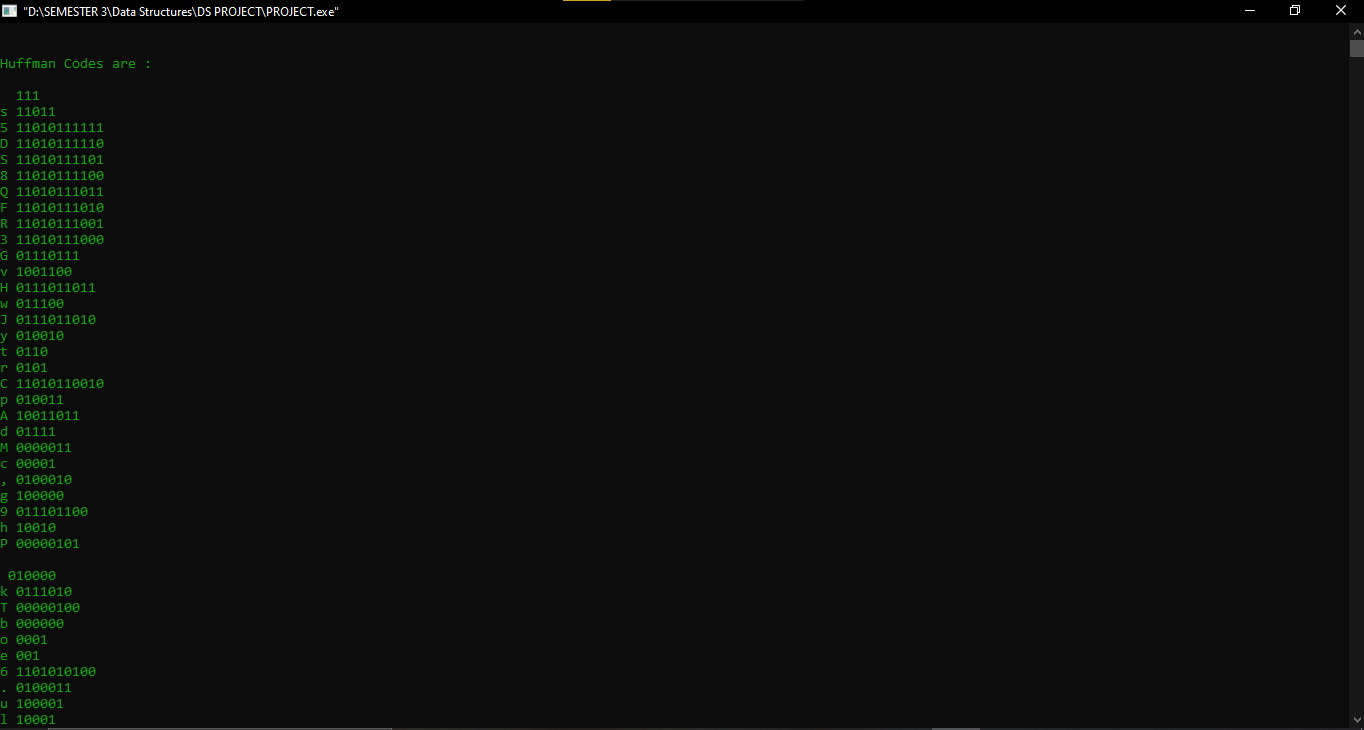
Snippets:

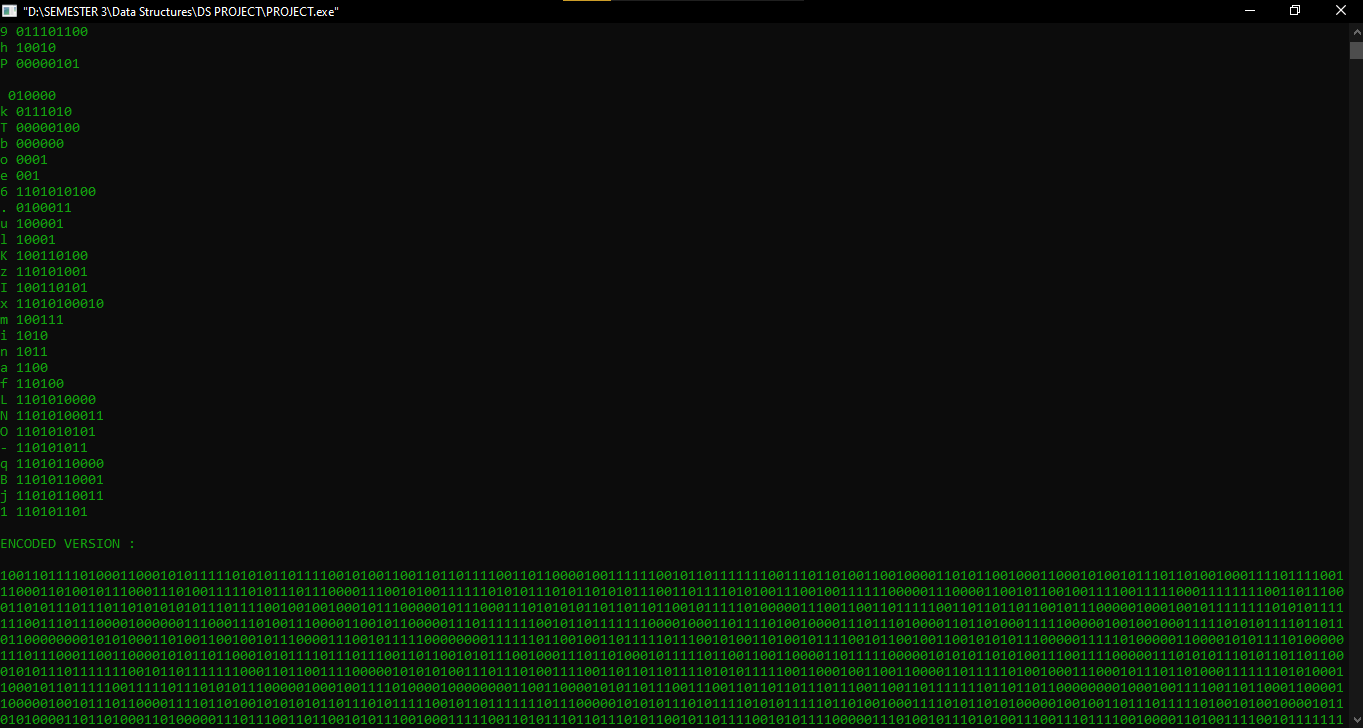
Input:





Output:







Data Structures is used: Trees, Priority queue , min heap, recursion.

Others: unordered\_map, pair order, vector and etc.