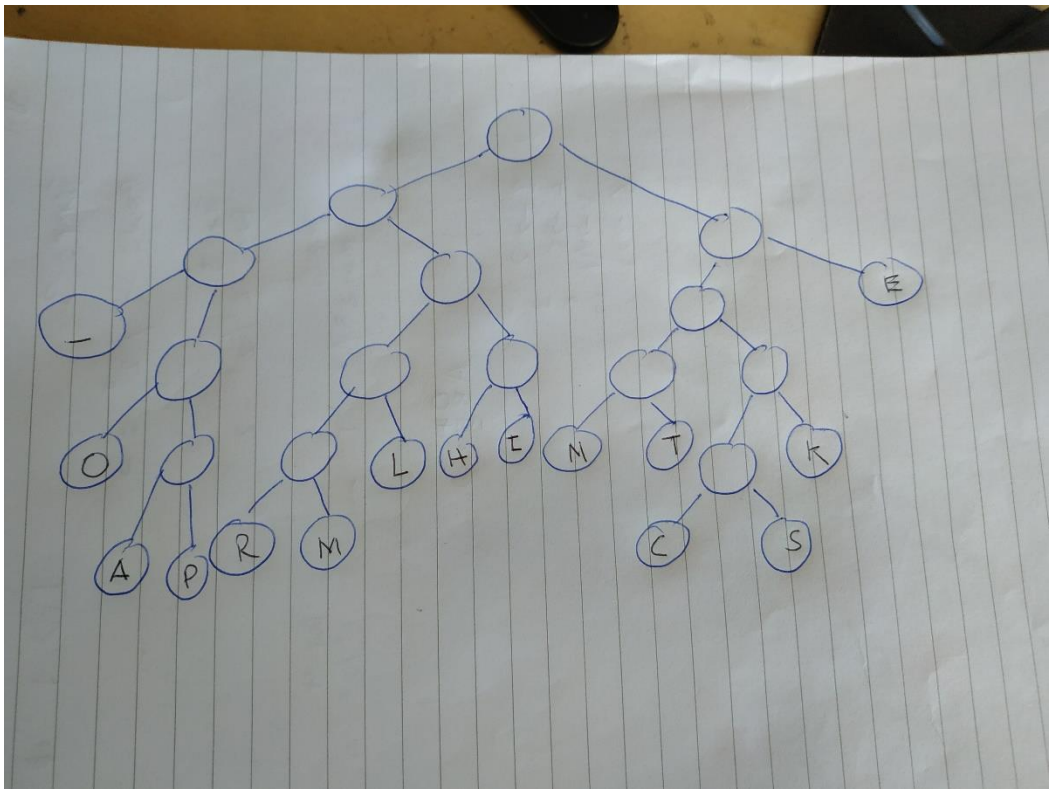
Lleno Anya
18357493

Task 1 AND 3 Compression assignment COMP20290

TASK 1

Character	Frequency	Codeword
T	1	1001
H	2	0110
E	5	11
R	1	01000
SPACE	5	000
I	2	0111
S	1	10101
N	1	1000
O	2	0010
P	1	00111
L	2	0101
A	1	00110
C	1	10100
K	1	1011
M	1	01001



Lleno Anya
18357493

Compression Analysis

File	Compression ratio	Compression Time	Decompression Time	Decompressed File Size
Genome	3.97	6.89ms	4.32ms	6251b
Med Tale	1.86	5.99ms	3.38ms	5734b
MobyDick	1.76	94.4ms	50.9ms	1,213,621 b
Q32x48	1.88	.879ms	.683ms	192b
Board	1.83	7.14ms	4.17ms	9023b

Q3)

Compression of an already compressed file is equal in size to the initial compression.

This is because making a Huffman trie of the bits frequency will return the initial Huffman trie with bit strings as the leaf nodes instead of characters. The resulting trie will be the same in structure.

Q4)

My Huffman implementation has a higher compression ratio that the Run Length Algorithm 1.88 > 1.34.

This can be explained by how each compresses data. Run length compresses data by reporting the number of consecutive data elements that are the same. This means that if the same data elements are separated by other data, they are recorded separately. Huffman coding on the other hand, compresses all similar data elements uniquely, i.e., there is only one code for every occurrence of a character. This allows more compact compression to be achieved