CMPE 365 | Algorithms I

Lab 7 and 8 Submission

November 2nd, 2019

Daniyal Maniar | 20064993

Part 1

The code-string dictionary generated is the following:

ASCII Code	Huffman Code
101	000
105	0010
115	0011
110	0100
10	01010
80	01011000
113	0101100100
34	010110010100000000000000000000000000000000000000000000000000
35	010110010100000000000000000000000000000000000000000000000001
36	010110010100000000000000000000000000000000000000000000000001
37	010110010100000000000000000000000000000000000000000000000001
38	010110010100000000000000000000000000000000000000000000000001
40	010110010100000000000000000000000000000000000000000000000001
41	010110010100000000000000000000000000000000000000000000000001
42	010110010100000000000000000000000000000000000000000000000001
43	010110010100000000000000000000000000000000000000000000000001
47	010110010100000000000000000000000000000000000000000000000001
48	010110010100000000000000000000000000000000000000000000000001
49	010110010100000000000000000000000000000000000000000000000001
50	010110010100000000000000000000000000000000000000000000000001
51	010110010100000000000000000000000000000000000000000000000001
52	010110010100000000000000000000000000000000000000000000000001
53	010110010100000000000000000000000000000000000000000000000001
54	010110010100000000000000000000000000000000000000000000000001
55	010110010100000000000000000000000000000000000000000000000001
56	010110010100000000000000000000000000000000000000000000000001
57	010110010100000000000000000000000000000000000000000000000001
60	010110010100000000000000000000000000000000000000000000000001
61	010110010100000000000000000000000000000000000000000000000001
62	010110010100000000000000000000000000000000000000000000000001
64	010110010100000000000000000000000000000000000000000000000001
70	010110010100000000000000000000000000000000000000000000000001
74	010110010100000000000000000000000000000000000000000000000001
75	010110010100000000000000000000000000000000000000000000000001
81	010110010100000000000000000000000000000000000000000000000001
86	010110010100000000000000000000000000000000000000000000000001
88	010110010100000000000000000000000000000000000000000000000001
90	010110010100000000000000000000000000000000000000000000000001
91	010110010100000000000000000000000000000000000000000000000001
92	010110010100000000000000000000000000000000000000000000000001
93	010110010100000000000000000000000000000000000000000000000001
94	010110010100000000000000000000000000000000000000000000000001
95	010110010100000000000000000000000000000000000000000000000001
96	010110010100000000000000000000000000000000000000000000000001
120	010110010100000000000000000000000000000000000000000000000001
122	010110010100000000000000000000000000000000000000000000000001
123	010110010100000000000000000000000000000000000000000000000001
124	010110010100000000000000000000000000000000000000000000000001
125	010110010100000000000000000000000000000000000000000000000001
126	010110010100000000000000000000000000000000000000000000000001
33	010110010101
39	01011001011
66	0101100110
68	01011001110
106	01011001111
118	0101101
109	010111

The above results and the decoded files can be seen in the “Part1” directory

Part 2

Based upon testing the best Canonical Collection to use to generate the codes is “Canonical Collection 3”. We can see that Collection 3 contains the most characters, thus getting the most accurate results for character frequency. This result makes sense since Huffman coding relies upon generating the smallest codes for the most used characters. If we do our best to see which characters are the most used, then we can get the best outcome by generating the shortest encoding. The results for the size comparisons can be seen below.

```
Canonical Collection 1 creates a larger sized Earth.txt encoded file.
Encoded: 1074701 bytes Original: 448780 bytes.
Canonical Collection 1 creates a larger sized Mystery.txt encoded file.
Encoded: 1174466 bytes Original: 453904 bytes.
Canonical Collection 1 creates a larger sized Myths.txt encoded file.
Encoded: 2104239 bytes Original: 740678 bytes.
Canonical Collection 1 creates a larger sized Simak.txt encoded file.
Encoded: 807322 bytes Original: 317530 bytes.
Canonical Collection 1 creates a larger sized Wodehouse.txt encoded file.
Encoded: 1047270 bytes Original: 412564 bytes.
Canonical Collection 2 creates a smaller sized Earth.txt encoded file.
Encoded: 254603 bytes Original: 448780 bytes.
Canonical Collection 2 creates a smaller sized Mystery.txt encoded file.
Encoded: 264030 bytes Original: 453904 bytes.
Canonical Collection 2 creates a smaller sized Myths.txt encoded file.
Encoded: 497415 bytes Original: 740678 bytes.
Canonical Collection 2 creates a smaller sized Simak.txt encoded file.
Encoded: 183401 bytes Original: 317530 bytes.
Canonical Collection 2 creates a smaller sized Wodehouse.txt encoded file.
Encoded: 235156 bytes Original: 412564 bytes.
Canonical Collection 3 creates a smaller sized Earth.txt encoded file.
Encoded: 251596 bytes Original: 448780 bytes.
Canonical Collection 3 creates a smaller sized Mystery.txt encoded file.
Encoded: 259137 bytes Original: 453904 bytes.
Canonical Collection 3 creates a smaller sized Myths.txt encoded file.
Encoded: 453738 bytes Original: 740678 bytes.
Canonical Collection 3 creates a smaller sized Simak.txt encoded file.
Encoded: 179546 bytes Original: 317530 bytes.
Canonical Collection 3 creates a smaller sized Wodehouse.txt encoded file.
Encoded: 231141 bytes Original: 412564 bytes.
The best Canonical Collection to encode the files with is 3
```

The results of each of the Canonical Collection encoding/decoding can be seen in “Part2/Data/”. Additionally, the “HuffmanCoding_16mdm13_DaniyalManiar.py” script outputs the above results with the minimum encoded bits.