Assignment #4

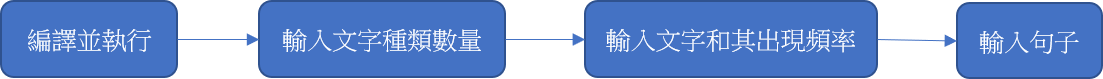
1. Build guide

比較Huffman coding和Arithmetic coding的源碼執行時所需的系統執行時間。

1. Execute guide

因為源碼使用C++進行編寫，因此編譯時需輸入g++ HW4.cpp -o main，接著輸入./main執行，一開始會要求輸入欲壓縮的文字種類數量，接著需輸入文字及其出現頻率，接著會顯示經過Huffman coding後，各文字壓縮後的結果與系統執行時間，以及Arithmetic coding所需要的各文字出現機率(由前面輸入的頻率計算而得)與區間，接著要輸入欲壓縮的句子(須由前面輸入的文字組成)，最後顯示壓縮後的結果與系統執行時間。

1. Flow chart



1. System structure
2. 詢問文字種類數量。
3. 詢問文字及其頻率。
4. 計算各文字的機率及區間。
5. 開啟計時器測試Huffman coding。
6. 將各文字建立成一個node。
7. 將node以頻率由小排至大。
8. 提取頻率最小的兩個node，並從array中移除。
9. 建立新的node，其左兒子與右兒子分別為7.中提取的最小及第二小的node，頻率為兩者相加。
10. 將node插入array中。
11. 重複6.~9.，直到node剩一個。
12. 用遞迴的方式拜訪樹，先往左，再往右，往左為零，往右為一，走到葉子後，按照順序印出。
13. 結束計時器並顯示系統執行時間。
14. 用表格顯示文字及其機率與區間上下值。
15. 詢問與壓縮句子。
16. 開啟計時器測試Arithmetic coding。
17. 依照句子中文字出現的順序更新各文字區間上下值，直到句子結束。
18. 結束計時器。
19. 顯示句子壓縮後的結果。
20. 顯示系統執行時間。
21. Source code
22. Huffman coding:

https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/?fbclid=IwAR2yLbUGOsTphC2B5MDwojk0q\_NsvV6Kg2RHd7xZPKrNcN4a6OYtxUypVBg

1. Arithmetic coding:

https://gist.github.com/mohitbansal964/3a40d80be9c78f1ac267ac329391c545?fbclid=IwAR0C1m\_2uiXP9xbMw\_ivY1UA4dYaMqx9tJxrol6Bje6MTD6pT7SMtx\_m0LA