Huffman-based Text Encoder and Decoder

Home of a simple Huffman-based encoder and decoder.

Why

This project was requested by Professor Diego V. Noble in my BCompSc university course.

The project

Given the King James Bible, encoded in ASCII, run the encoder.
The encoder will create a new file, `king_james_ENC` (no extension), with a one-line header containing the character decoding table and the encoded text. The decoder reads this file and uses the decoding table to decode the contents of the encoded file, saving it in `king_james.txt`.

I used a Priority Queue for the final version, but I also made a version that uses a Vector. This was done to emphasize the difference in execution time between the two data structures.

Using a priority queue, we have close to $O(nlog(n))$ in complexity. Using a vector, we have $O(n^2)$ complexity.

Execution times

The program was executed 20 times, half of which used vectors and the other half used a priority queue to build the Huffman Tree.

It's worth mentioning that the file I used (king_james.txt) is around 4MiB and 100224 (one hundred thousand two hundred twenty-four) lines, which resulted in fake results due to the small elapsed times (resulting in an also small standard deviation). The results may have been tainted by random function calls by the processor (e.g. background processes).

In order to get more accurate results, this program should be run using a bigger text file (e.g. the Wikipedia Front Page).

Executing using a vector

Execution no.	Elapsed time (in microseconds)
1	194
2	188
3	188
4	186
5	189

6	188
7	188
8	194
9	187
10	194
sum	189.6

Executing using a Priority Queue (Min Heap)

Execution no.	Elapsed time (in microseconds)
1	24
2	27
3	28
4	27
5	28
6	27
7	28
8	29
9	28
10	28
sum	27.4

Results

After 10 executions each:

Data Structure	Elapsed Time (μ)
Vector	189.6 microseconds (0.0001896 seconds)
Priority Queue	27.4 microseconds (0.0000274 seconds)

Even with tainted results, it's clear to see that using a priority queue is much faster than using a std::vector.