Huffman Coding

The following generates a Huffman code for any given probability distribution:

**INPUT:** An input probability distribution $P$ for a set of symbols $S$

**OUTPUT:** A binary tree encoding $P$

**FOR EACH** symbol $s$ in $S$:  
create a tree leaf node holding $(P(s), s)$;

**PLACE** all nodes into a priority queue $Q$;

**WHILE** $Q$ contains more than one element:  
remove the two nodes $s_1, s_2$ with the smallest probabilities from $Q$;  
create a new node with $s_1, s_2$ as children and $P(s_1) + P(s_2)$ as probability;  
add the new node to the queue;

RETURN the code tree $T$; (its root is the last node left in $Q$)

1. Implement this functionality in your programming language of choice.

2. Write a method to access the codes for a given symbol. You will have to create a mapping $symbols \rightarrow codewords$. For this, do a tree traversal and record the path leading to each leaf symbol. The path is a sequence of $b \in \{0, 1\}$, 0 for every left branch and 1 for every right branch you take on the way.

3. Use the code and the data for the last exercise. This time determine the probability distribution of words. Compute the Kullback-Leibler divergence as in the last exercise. To avoid division by zero, you must smooth the second distribution. Please smooth the second distribution using add-1 smoothing*.

4. For the Maximum Likelihood estimate you obtained for words in text1 generate the Huffman code. Encode text1. Then encode the intersection** of text1 and text2 with the same code.

5. Discuss length and entropy of the encoded texts. Refer to limits of the optimal code, compression with a mismatched code and the expected length of the encoded text.
6. BONUS for Monday tutorial: Create a message with a non-uniform distribution of symbols that meets the lower bound of the Coding Theorem with equality.

7. BONUS for Wednesday tutorial: Generate a Huffman code $C$ for the character distribution $P$ of text1. Encode the text and measure the average codeword length $L$. Now create a distribution $Q$ of characters such that $p(c_i) = 2^{-l_i}$, where $l_i$ is the length of the $i$-th codeword $c_i \in C$. Compare $D(P||Q)$ with $L$ and $H(P)$. Do you notice anything interesting? How do you explain your observation? Make sure that all measurements are in bits (use base-2 logarithm).

* In order to add-1 smooth the probability distribution of text2, simply increment all counts made for text2 by 1. Then add 1-counts for all observations that are in text1 but not in text2. The probabilities should be based on the new counts.

** In order to determine the 'intersection' of the two texts, simply ignore all words in text2 that are not in text1.

---

**Note on Submission** Please use PDF as a document format. If you need to compress files, use ZIP or GZIP.

If you attend the **Mon. tutorial**, the **deadline** is **Friday 23th May, 23:59**. In that case send the solutions to s9lshack@stud.uni-saarland.de.

If you attend the **Wed. tutorial**, the **deadline** is **Sunday 25th May, 23:59**. In that case send the solutions to kalofoli@csid.upatras.gr.

Please indicate each group member’s name in the filename: Ex03-MemberName1(MN)-MemberName2(MN)-MemberName3(MN).zip