This is an individual CA. It is due on Friday 20 April. You may be asked to demo and explain your solutions after the Easter break. If you use code from any source you must comment and cite it correctly, otherwise you may get zero marks. *Ignorantia juris non excusat.* Here are examples of how to cite other peoples code http://elearning.it-tallaght.ie/mod/page/view.php?id=152811 and http://uark.libguides.com/content.php?pid=155080&sid=1780817

You are required to carry out the tasks below in order to develop an optimal **Huffman Tree** with which to compress a file of ASCII text. **You will be expected to make use of the relevant template classes in the STL.**

You should aim to complete the first section of this problem (i.e. the steps 1 through 5 below). Once you have completed tasks 1-5 sections complete the other sections 6 & 7.

The first 5 tasks will encode and decode the file into a string of 0's and 1's using a Huffman tree

1. Given a text file, determine the frequency of each character in the text (**map** of character and frequency).
2. Build an optimal Huffman tree to represent these characters with these frequencies (maintain a **priority queue of trees**, removing and joining trees until only one tree remains – the final Huffman tree required) Hint: each Hoffman Tree should have a weight data member, as well as a pointer to its root node.
3. Use the tree to map each character to the string of 0's and 1's needed to encode it (pre-order traversal of tree, store results in a **map** of character and string).
4. Use the map to encode the text to a string of 0's and 1's and write to an encoded file
5. Use the Huffman Tree to decode the text and write to another file.

Tasks 6 and 7 compress and un-compress the file:

6. To compress: break up the string of 0's and 1's into 8 bit chunks, and write the character represented by each chunk to the compressed file.
7. Decode by replacing each character with the 8 bits needed to represent it.