Huffman Code Trees
CS101 Project 5
Due 11/23

You are to write a program that reads a text file, computes a Huffman code tree for that text file, and writes out the encoded version of the text.

- Your program should read the text from the file given as a command-line argument.
- You should compute the number of occurrences of each character in the file, and each character (and its frequency) should be placed in a new tree node.
- You should build a min-heap containing these nodes.
- Build the Huffman Code Tree using the heap. As you create new internal nodes, give them a unique integer label.
- Write the pre-order traversal followed by the in-order traversal of the Huffman code tree to the file "tree.txt". If the node is an internal node, write I:xxx, where xxx is integer label, and write L:xxx where xxx is the ASCII value of the character otherwise. Separate the nodes by a space, the two traversals should be separated by a newline.
- Construct a table containing the encoding for each character, storing the encoding as a string.
- Encode the original text, writing the encoded version to "code.txt". This file should be ASCII '0' and '1' characters (much easier to debug)
- (for 5 bonus points) also create a true binary version of the encoded text, writing it to "code.bin". In the binary file, use the first two bytes to indicate the number of characters in the text. If the last character doesn't finish a byte then pad the last byte with 0's.
- One additional output to generate is the file "length.txt". For every character that occurs at least once in the text, output the ASCII value and the length of the encoding.

- (For 20 extra points) Write a program that reads the output files above, tree.txt and code.txt, outputting the original text.

Requirements:

- You should build all the data structures that you use yourself. You must create a Heap data structure that uses an array and implements insert and extract-min that run in O(lg N) time.
- Your makefile should build the executable named "encode". If you do the 20 point bonus, the same makefile should also build "decode".
- Zip all of your source code and makefile into a single .zip file for submission.
- You must use good object based organization, i.e. use classes in an appropriate way.
- Because the semester is coming to a close, no assignments will be accepted past one day late (for a 10% penalty) 11/24.

Example
Suppose that the file “foo.txt” contains the following text:
ALLALABAMAFOOTBALL

Then executing:
encode foo.txt
should produce output files such as the following. Note that this is only an example of a correct output. The tree and codes produced by your program would likely be different.

tree.txt contains:
I:505 I:504 L:076 L:065 I:503 I:502 L:066 L:079 I:501 I:500 L:084 L:070 L:077
L:076 I:504 L:065 I:505 L:066 I:502 L:079 I:503 L:084 I:500 L:070 I:501 L:077

code.txt contains:
01000001000110001111011101101101100100010000

length.txt contains:
065 2
066 3
070 4
076 2
077 3
079 3
084 4