Foundations of Computer Science Exam 1 **Due: 12/11/2023**

Instructions:

This will be a take-home programming assignment.

1. You will use the encoding code provided and add the following two parts:

Assigning code:

- You will create a sample text file, raw.txt to run through the encoding. (Be creative) Generate a table showing how much the file was compressed. (See example on next page.)
- Using the provided HuffmanTree code, you will add code to encode each symbol in the raw.txt to have a binary coding. Each left branch is a 0, and each right branch is a 1. Output the encoding of the raw.txt into a file called encoded.txt.

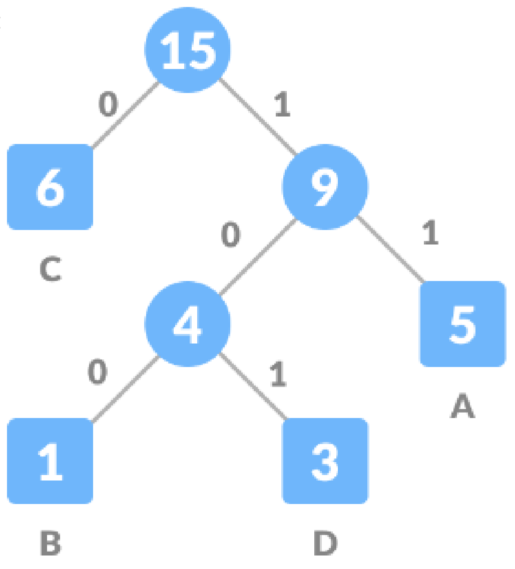
Implementing decoding:

- You will trade encoded text files with the person assigned to you (via email), and decode their text file. Save your decoded text file as decoded.txt. Please provide your partner with the character, and frequency inputs as two lines of data in a file counts.txt (formatted like the example), but not the original raw.txt. You should run the program to build the Huffman tree and run the decode on the encoded.txt.
- Details on decoding:
  1. Start at the root:
  2. For each bit in the encoded message:
     a. If the bit is 0, it moves to the left child of the current node.
     b. If the bit is 1, it moves to the right child of the current node.
     c. When a leaf node is reached, the character associated with that leaf node is considered decoded.
  3. Reset to root
  4. This process repeats until the entire encoded message is decoded, and the original message is reconstructed.

- Give some observations and/or explanations you think that make Huffman coding give you back the original message every time, without any mistakes and/or guessing at where a new symbols encoding starts.

Submission: Please submit all code as .java, and submit the text files via email using the subject line **FOCS: test 2**. Make sure to use reply all in your text file exchange.

Example:
Input text: BCAADDDCCACACAC
Huffman Tree:



Encodings:
A : 11
B : 100
C : 0
D : 101

Encoded text: 1000111110110110100110110110

Summary Table:

Character	Frequency in string	Assigned Code	Size
B	1	**100**	1 x 3 = 3 bits
D	3	**101**	3 x 3 = 9 bits
A	5	**11**	5 x 2 = 10 bits
C	6	**0**	6 x 1 = 6 bits
4 x 8 = 32 bits	Total = 15 bits		Total = 28 bits

Without Compression: 120 bits needed (8 bits * 15 characters)
With Compression: 75 bits needed (32 bits + 15 bits + 28 bits)