CpSc 2120 — Goddard — Spring'18

Lab 8: Huffman Code

Wendy wanted to implement Huffman coding. She found code for a Heap on a friendly neighborhood course website. She proceeded to:

- Create a new class CharRecord;
- Adapt the Heap code; and
- Produce the huffman code.

The resultant HuffHeap and huffman files (header and implementation) are provided. These should **NOT** be changed. Your task for this lab is:

1. Produce a suitable CharRecord class.
   (Hint: look through the code for HuffHeap and huffman.cpp to see what they assume about CharRecord.)
2. Then compile and run huffman to prove that it works. Note that huffman takes an upper-case string on the command-line. (You should get output like below.)

Submit via handin just your CharRecord files.

Though a full understanding of the huffman code is not needed for the lab, here is some explanation. It works in two stages. It first builds the tree, storing the tree as a string. For example, the string `((A:B):C)` represents a tree with three leaves: the left child of the root is an internal node that has nodes `A` and `B` as children; and the right child of the root is a leaf with `C`. Then the code uses the tree to print out the resulting encoding.

```
hornet4.cs.clemson.edu[177] ./a.out ENDLESSNESS
D : 1
E : 3
L : 1
N : 2
S : 4

(D:L) : 2
((D:L):N) : 4
(E:((D:L):N)) : 7
(S:(E:((D:L):N))) : 11

S -> 0
E -> 10
D -> 1100
L -> 1101
N -> 111
```