List of supporting files

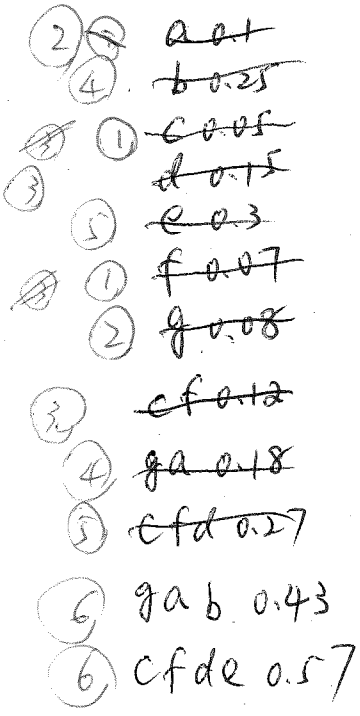
1. **data2.txt:** input data file;
2. **nodedata.h** and **nodedata.cpp**: NodeData class;
3. **lab2.cpp**: containing main(), to help clarify the functional requirements;
4. **lab2output**: correct output in using **lab2.cpp**;
5. **classAndSideway.txt**: structure of BinTree class and 2 functions for helping display a binary tree as if you are viewing it from the side. *NOTE:* They will be part of your program. You can adjust this definition to your implementation (although you must have the three pointers in the Node and it must work with **lab2.cpp**). Node could be a class and it doesn't necessarily have to be nested in the BinTree class. If a class, it is acceptable for the BinTree to be a *friend* of Node.

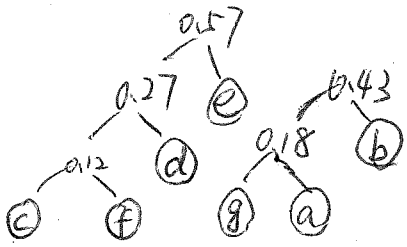
Submission Requirements:
All the rules, submission requirements, and evaluation criteria are the same as the assignment 1.

Part 2. Use as many 8½ by 11 inch pieces of paper, one-sided, to make your work clear.
1. Use Huffman coding to encode these symbols with given frequencies:
a: 0.10 b: 0.25 c: 0.05 d: 0.15 e: 0.30 f: 0.07 g: 0.08

2. Build (draw) a heap (minheap, smallest value at root) by inserting one item at a time with the values:
15 20 3 4 8 10 9 5 14 2 25

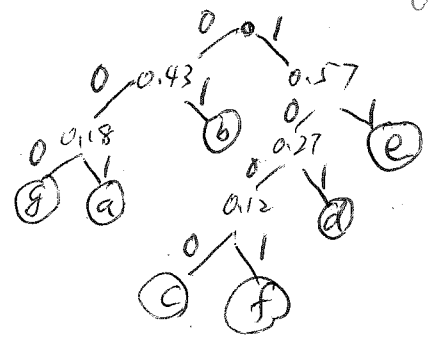
3. Build (draw) a heap (minheap, smallest value at root) using the linear algorithm with the values:
10 2 8 6 20 3 5 12 4 10 15





left small
right big
before merge,
compare

switch



a 001
b 01
c 1000
d 101
e 11
f 1001
g 000