File Compression

Monday, April 3, 2023 2:30 PM

"Make the common case small"

Keep in realm of ASCII Characters
- All characters take the same amount of space
- Change the representation so it reflects the usage of each character

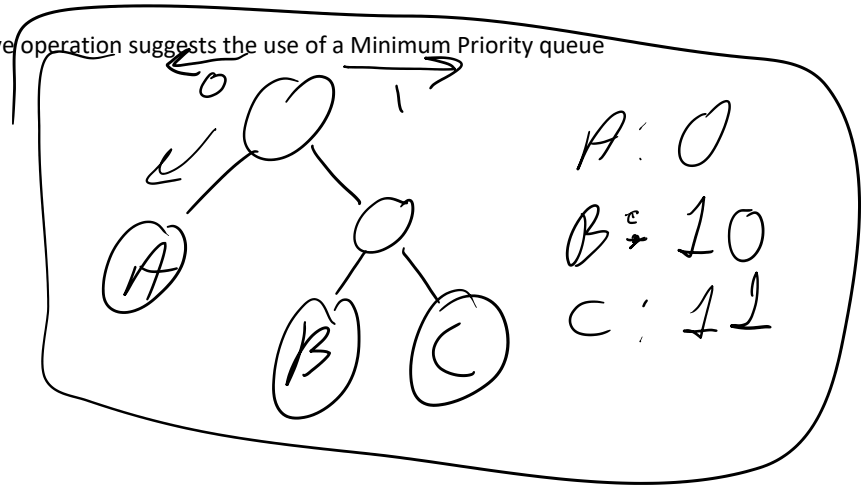
Perform a "frequency count"
Build the bit-by-bit representation based on the frequency count

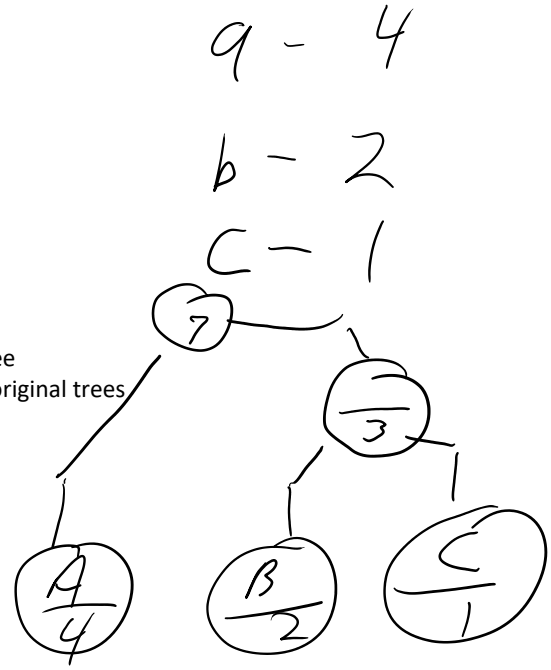
Huffman Encoding

Create a "forest" of binary trees
Initially each tree represents a single character and the frequency count for that character

Select the two tree with smallest frequency count and merge them together into a single tree
- the frequency count of the resulting tree is the sum of the frequency counts of the two original trees

The above operation suggests the use of a Minimum Priority queue





Assume we have the following "encoded" message: 0101101000

That message is really: ABCABAA

What is the difference between "char" and "unsigned char" in C/C++?
char: range of values is -128 to 127
unsigned char: range of values is 0 to 255

```
unsigned char ch;

 cin >> ch;

file >> ch;
```

What is the difference between "int" and "unsigned int" in C/C++?

How do I write out information bit-by-bit?

```
boolean writeBit( boolean bit,   outstream stream    ) {
   static int bitpos = 0;
   static unsigned char bit = 0;
```

```
  if (bit)
      bit = 1 | (bit << 1);
  else
      bit = 0 | (bit << 1);
 bitpos++;

 if (bitpos == 8) {
    stream << bit;
    bit = 0;
     bitpos = 0;
    return true;
  }
 return false;
}
```

4 Operations for Project 4:

1. Create the Huffman Information File .hi
    a. contains 128 lines of information (one for each ASCII character)
    b. decimal value of the ascii character
    c. the huffman code for that character - characters of 0's and 1's
    d. Input for this operation is an ASCII character file.

2. Load a .hi file to use with Compression/Decompression
    a. Build 2 items
    b. character to code vector - used during compression
        i. given in the .hi file
    c. Huffman Tree - used during decompression

3. Compress a file
    a. given a ASCII file
    b. need to have a .hi file loaded
    c. output is a .hc
    d. output is written "bit-by-bit" need to handle writing the last byte of information

4. Decompress a file
    a. given a .hc file
    b. need to have a .hi file loaded
        i. if a different .hi is used for decompression than for compression, the results should be meaningless
    c. read the .hc file "bit-by-bit", until the last byte is read in
        i. for each bit, traverse the Huffman Tree (on 0 go left; on 1 go right)
        ii. when a leaf node is found, a ASCII Character has been decompressed
        iii. Start at the root of the tree for the next bit