The idea in this assignment is to implement Huffman encoding, a method for compressing text files into a fraction of their original size. Standard ASCII code uses seven bits to store each character in a text file, so common characters like blanks and vowels use the same number of bits as uncommon characters like ‘X’ and ‘~’. Huffman encoding associates a small sequence of bits with characters that occur frequently in a text file, and substantially larger bit sequences with the characters that are less common. In this way, an entire text file is usually much shorter when it is Huffman encoded than when it is merely ASCII encoded.

For example, consider the following piece of text:

In the day we sweat it out in the streets of a runaway American dream.

Notice that the characters in this text occur with the following frequencies:

<table>
<thead>
<tr>
<th>Character</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
</tr>
<tr>
<td>c</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>8</td>
</tr>
<tr>
<td>h</td>
<td>2</td>
</tr>
<tr>
<td>i</td>
<td>3</td>
</tr>
<tr>
<td>n</td>
<td>4</td>
</tr>
<tr>
<td>r</td>
<td>4</td>
</tr>
<tr>
<td>t</td>
<td>7</td>
</tr>
<tr>
<td>w</td>
<td>3</td>
</tr>
<tr>
<td>blank</td>
<td>1</td>
</tr>
</tbody>
</table>

The Huffman encoding procedure begins by placing each of these character/frequency pairs into a single node binary tree:

Next, the procedure merges together those trees with the smallest frequency values, placing the sum of the frequencies in the new root node (with a null character value), and continuing until all of the trees have been merged into a single larger binary tree.
Merging \((yAcouhrm, fIisw<blank>)\) and then \((yAcouhrmfIisw<blank>, at.dne)\):

The actual Huffman encoding for each character is obtained by traversing the Huffman tree from the root to the character, concatenating a zero-bit every time a left offspring is traversed and a 1-bit every time a right offspring is traversed:

<table>
<thead>
<tr>
<th>Character</th>
<th>Huffman Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10010</td>
</tr>
<tr>
<td>a</td>
<td>000</td>
</tr>
<tr>
<td>c</td>
<td>100011</td>
</tr>
<tr>
<td>d</td>
<td>01001</td>
</tr>
<tr>
<td>e</td>
<td>011</td>
</tr>
<tr>
<td>f</td>
<td>1100000</td>
</tr>
<tr>
<td>h</td>
<td>10110</td>
</tr>
<tr>
<td>i</td>
<td>11001</td>
</tr>
<tr>
<td>I</td>
<td>110001</td>
</tr>
<tr>
<td>m</td>
<td>10111</td>
</tr>
<tr>
<td>n</td>
<td>0101</td>
</tr>
<tr>
<td>o</td>
<td>10010</td>
</tr>
<tr>
<td>r</td>
<td>1010</td>
</tr>
<tr>
<td>s</td>
<td>11010</td>
</tr>
<tr>
<td>t</td>
<td>001</td>
</tr>
<tr>
<td>u</td>
<td>10011</td>
</tr>
<tr>
<td>w</td>
<td>11011</td>
</tr>
<tr>
<td>blank</td>
<td>111</td>
</tr>
</tbody>
</table>

Note that the original 70-character message would be 490 bits long in ASCII, but turns out to be only 276 bits long using this Huffman encoding. Also note that traditional "white-space" characters (blanks, tabs, and line-feeds) are considered characters in this programming assignment.

Your assignment is to implement Huffman encoding for the text in a user-specified input file. Your driver program will read the input file and create an array of input characters and their corresponding frequencies within the text file. This array will then be used to generate a linked list of one-node binary trees, each of which will contain a character and its integer frequency from the text file.

Next, your driver will repeatedly remove the two "smallest" trees from the linked list and merge them into a taller tree by creating a new root (with the null character and the sum of their frequencies) and placing the two removed trees into the new tree as left and right subtrees. This new tree is then inserted back into the linked list, and the process continues until the linked list contains only one tree, the final Huffman tree.

At this point, your driver will output the Huffman encodings for all of the characters by recursively traversing the entire tree, outputting the Huffman codes for each leaf node into a user-specified output file. This is illustrated below for a sample text file, `Grinch.txt`, which is available on the course website.

You'll need at least three classes to implement this assignment: a linked list class template (you'll be using a linked list of binary trees), a binary tree class template (you'll be using a binary tree of character/frequency pairs), and a character/frequency-pair class.

The linked list will need the usual member functions (default constructor, copy constructor, destructor, \texttt{getNode}, \texttt{isEmpty}, \texttt{insert}, and \texttt{remove}), but additional member functions will probably be useful as well (e.g., \texttt{removeSmallest}, to remove the smallest element of the list; \texttt{size}, to count the number of elements in the list, etc.).

The binary tree will also need its usual member functions (default constructor, copy constructor, destructor, assignment operator, \texttt{getNode}, \texttt{isEmpty}, and \texttt{insert}), but additional member functions will definitely be needed here, including a less-than operator (that compares root node inequality), a recursive output routine (to traverse the tree and output the Huffman code for each leaf node), and routines to reset the root's value, its left pointer and its right pointer (to facilitate the merger of two binary Huffman subtrees).

The character/frequency-pair class will need an output operator (which will be used when outputting the Huffman codes) and a less-than operator (which will be based upon a comparison of a pair's relative frequencies).

The format of the output for the `Grinch.txt` input is illustrated on the next page.

Include explanatory comments that document the purpose of each file and function, and make certain that your program exhibits modularity, modifiability, and readability.
He took the Whos' pudding!
He took the roast beast.
"It's easier than it looks as quick as a flash."
Why, that Grinch even took their last can of Who-bash!
Then he stuffed all the food up the chimney with glee.
And HOW!"
"I will stuff up the tree!"
And the Grinch grabbed the tree, and he started to shove
He turned around fast, and he saw a small Who!
Little Cindy-Lou Who, who was not more than two.
The Grinch had been caught by this tiny Who daughter
Who got out of bed for a cup of cold water.
For he knew every Who down in Whoville beneath
"Santy Claus, why,"
"Why are you taking our Christmas tree? WHY?"
She stared at the Grinch and said,
"There's a light on this tree that won't light on one side."
"I'll fix it up there. Then I'll bring it back here."
Then he paused.
"I've never seen a Grinch do such a thing!"
"Then he patted her head.
"And he got a drink and he sent her to bed.
"And when Cindy-Lou Who went to bed with her cup,
"He went to the chimney and stuffed the tree up!"
The Grinch put his hand to his ear.
So he paused.
"What a great Grinchy trick!"
And he chuckled, and clucked,
"An awful idea!"
Then he got an idea!
"...But HOW?"
I MUST stop this Christmas from coming!
And the Grinch thought of something he hadn't before!
He had an idea!
And he brought back the toys!
And what happened then...?
"Maybe Christmas...perhaps...means a little bit more!"
Then he shook!
"This sound sounds merry!"
But the sound wasn't sad!
And the Grinch put his hand to his ear.
And he did hear a sound rising over the snow.
And the Grinch, with his grinch feet ice cold in the snow,
Standing on Christmas Eve, hating the Whos,
May have been that his heart was two sizes too small.
No one quite knows the reason.
Now, please don't ask why.
But the Grinch, who lived just north of Who,
"All I need is a reindeer..."
"What a great Grinchy trick!"
And he chuckled, and clucked,
And he made a quick Santy Claus hat and a coat.
"I must stop this whole thing!
So he paused.
"I must stop this Christmas from coming!"
Then he growled, with his Grinch fingers nervously drumming,
"And they're hanging their stockings! he snarled with a sneer
For he knew every Who down in Who
Staring down from his cave with a sour, Grinchy frown
He stood there on Christmas Eve, hating the Whos,
"The Grinch look"
"All I need is a reindeer..."
I look just like Saint Nick!"
"And the Grinch thought, "I must stop this Who"
And they'd sing! SING! SING! SING! SING! SING! SING! SING!
What he saw was a shocking surprise!
Then he shook!
And the Grinch grabbed the tree, and he started to shove
He turned around fast, and he saw a small Who!
Little Cindy-Lou Who, who was not more than two.
The Grinch had been caught by this tiny Who daughter
Who got out of bed for a cup of cold water.
For he knew every Who down in Whoville beneath
"Santy Claus, why,"
"Why are you taking our Christmas tree? WHY?"
She stared at the Grinch and said,
"There's a light on this tree that won't light on one side."
"I'll fix it up there. Then I'll bring it back here."
Then he paused.
"I've never seen a Grinch do such a thing!"
"Then he patted her head.
"And he got a drink and he sent her to bed.
"And when Cindy-Lou Who went to bed with her cup,
"He went to the chimney and stuffed the tree up!"
The Grinch put his hand to his ear.
So he paused.
"What a great Grinchy trick!"
And he chuckled, and clucked,
"An awful idea!"
Then he got an idea!
"...But HOW?"
I MUST stop this Christmas from coming!
And the Grinch thought of something he hadn't before!
He had an idea!
And he brought back the toys!
And what happened then...?
"Maybe Christmas...perhaps...means a little bit more!"
Then he shook!
"This sound sounds merry!"
But the sound wasn't sad!
And the Grinch put his hand to his ear.
And he did hear a sound rising over the snow.
And the Grinch, with his grinch feet ice cold in the snow,
Standing on Christmas Eve, hating the Whos,
May have been that his heart was two sizes too small.
No one quite knows the reason.
Now, please don't ask why.
But the Grinch, who lived just north of Who, did NOT!