Colorado Mesa University
Dept. of Computer Science and Engineering

CSCI112 Data Structures

Practicum Week 10 - - - - *1% bonus (solution with no memory leaks)*
Due for demo in class on Friday April 2.

Aim:
To use a binary tree to analyze text.

Task:
One of the most interesting uses of computers has been to examine books purported to have been written by well-known authors and test them to see if they have certain properties associated with that author. One such analysis is word frequencies. This assignment is to produce a table of word frequencies of a piece of text by placing the words in a binary tree. For each word read, the tree is searched. If the word is not in the tree, the word is added to the tree and its frequency set to 1. If the word is already in the tree, its frequency is incremented. When the end of the text is reached, the tree is traversed and the frequencies printed.

As practice, we'll use the `string` class to read the words. See if you can use no character arrays at all.

To use the code for the class `BinaryTree` provided in the files `BinaryTree.h` and `BinaryTree.cpp`, we have to supply two things: a file called `DataType.h` which defines the type `DataType` to go into the tree; and a function

```
      int TreeDataCmp(const DataType&, const DataType&);
```

to compare two instances of `DataType`, to determine their relationship in the tree order. For our purposes, we need the words to be in alphabetical order. Firstly,

```
struct DataType
{
     string word;
     int freq;
};
```

The function just needs to compare the two instances' `word` member and return -1 if the first comes before the second in alphabetical order, 0 if they are the same word, and 1 otherwise.

Our program needs only follow this algorithm.

```
request a file name
open the file
read a word
while the file still has words to read
     construct a DataType instance with the word in it
```

```
        check to see if that word is in the tree
        if it is, increment its frequency
        if not, add it to the tree
        read the next word
close the file
traverse the tree, printing out the word and frequency
```

Five data files are provided, each containing excerpts of text. Ignore words that start with a digit (they are numbers) and make the first letter of each word lower case. An example of the form of the output is shown below for the first of the data files.

```
File to be analyzed: 10a.txt
aid     1
all     1
brown   1
come    1
dog     1
for     1
fox     1
good    1
is      1
jumps   1
lazy    1
men     1
now     1
of      1
over    1
party   1
quick   1
the     5
time    1
to      2
```

A point to ponder:
Many times in this exercise, whether it is in your code or the code provided, there are copies being made of `strings`. A `string` is a dynamic entity. All our copying is shallow. Is there memory being leaked? Have we reused memory which should not have been reused?

Demonstrate your completed code to peer tutors or myself by EOC Friday:

You may be asked to submit all files (tarball via email):