Homework 3 – Character Count

10 Points – Due November 12th

Description:

For this assignment you are going to create a program that allows a user to type in some text and then using a Standard Template Library Map object you will keep track of the individual characters and the number of times each character occurs in the user text. Then you will generate a table where the first column will display the characters from the user text, the second column will list the number of occurrences of each character in the user text, and finally the last column will list the percent that each character takes up in the entire user text that was entered.

Below is an example of running the program with the user entering "Hello happy world!"

```
Please enter a some text and press enter: Hello happy world!

Total letters in the user text = 18

Char      Count     Percent
---------------------------------
<SPACE>   2         11.11 %
!         1          5.56 %
H         1          5.56 %
a         1          5.56 %
d         1          5.56 %
e         1          5.56 %
h         1          5.56 %
l         3         16.67 %
o         2         11.11 %
p         2         11.11 %
r         1          5.56 %
w         1          5.56 %
y         1          5.56 %
```

The zip file you downloaded for this project contains a directory named **example** that contains some code the closely matches what is described in this write-up. You will modify the **main_map.cpp** file for this assignment. Loops and conditionals have been commented out so that the program will compile and execute when you get it and you will need to uncomment those areas when you start working on them. There are also numbers 1 - 10 in the comments for the sections you need to work on.

Maps

You might be wondering what exactly is this map thing that you are going to use for the program? A map contains pairs of data where one item is called a key and the other is called a

mapped value. Maps are sometimes also called dictionaries. In a dictionary you would look up a word to find the definition – the word is the key and the definition is the mapped value.

hap·py (hăp'ē)
adj. **hap·pi·er, hap·pi·est**
1. Characterized by good luck; fortunate.
2. Enjoying, showing, or marked by pleasure, satisfaction, or joy.
3. Being especially well-adapted; felicitous: *a happy turn of phrase.*
4. Cheerful; willing: *happy to help.*

Figure 1: Source: tfd.com - Accessed Nov 1st 2012

In Figure 1 the key is the word "happy" and the mapped value is the definition.

With a STL map object you can use any data type as the key and any data type as the mapped value, they do not have to be the same type. Below is an example of creating a map:

```
map<char, int> letters;
```

The map name is ***letters*** and it has a character key and an integer mapped value. To add values to the map there are several options and two are listed below.

```
letters['a'] = 4;
```

or

```
letters.insert(pair<char, int> ('a', 4));
```

Both methods will insert an element into the map that has a key of the character ***'a'*** and a mapped value of ***4***. Maps cannot contain duplicate keys, so if you put both of the statements above in a program it would result in only one new element being added to the map.

You can make an iterator for a map as well and it works similarly as the list and vector iterators.

```
map <char, int>::iterator iter;
```

Maps allow you to use the [ ] index syntax, just as a vector, string, and array do as well.

Accessing the data stored where an iterator is pointing is different because there is a pair of data and not a single value.

```
iter = letters.begin( );

cout << "K = " << iter->first << " V = " << iter->second << endl;
```

It is possible to find mapped values in a map by searching for the key. If the key is found an iterator pointing to the element is returned and can be used to then access the mapped value.

```
iter = letters.find('a');
```

```
cout << "Key \'a\' has a mapped value of " << iter->second << endl;
```

WARNING: If the key is not found the iterator returned by find points to the map end!

Below is a pictorial representation of a map with one element that has a char key and int mapped data. The iterator ***iter*** points to the first element of ***example_map*** and ***iter->first*** points to the key and ***iter-second*** points to the mapped value.

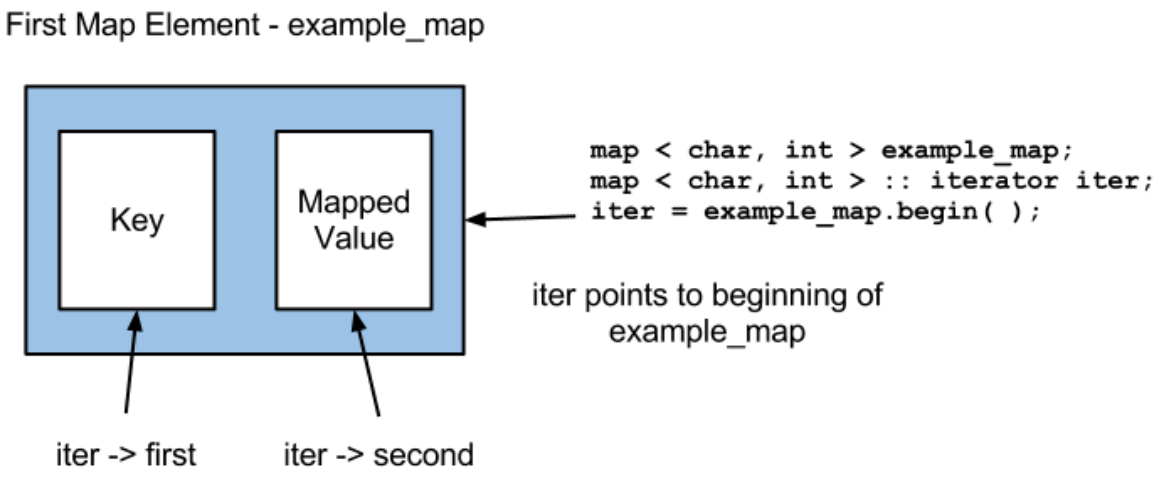

Figure 2: Pictorial Representation of a Map Element

For additional map information use this reference:
http://www.cplusplus.com/reference/stl/map/

Strings

A C++ string object is an array of characters that has several functions that you can use, such as size( ). A nice feature of strings is that you can iterate through them using [ ] like an array. Below is an example of printing out the individual characters of a string, each one on a separate line:

```
string course = "CSCI 1300";

for( unsigned int i = 0; i < course.size( ); ++i )
{
      cout << course[i] << endl;
}
```

If you have never seen the keyword **unsigned** used before, it means that the integer ***i*** will never contain a negative value, so the lowest value it will ever be is zero. The string ***size( )*** function returns an unsigned integer and if ***i*** is not the same type a warning is generated when the project is built. Your final project should contain no warnings!

What you need to do:

Once the user has entered their text into a string you will iterate through the string character by character and add them to the map, initially with a count of one. The character will be the key and the count of that character will be the mapped value. If the character is already in the map the previous count needs to be increased by one.

Now you need to iterate through the map to print out the values under each column. These values need to be lined up neatly. The space character and tab character need to display as <SPACE> and <TAB> respectively in the column for characters. The decimal points for the percentages should line up as in the example output. Here are several text formatting references to help you:

http://www.cplusplus.com/reference/iostream/ios_base/precision/
http://www.cplusplus.com/reference/iostream/ios_base/setf/
http://www.cplusplus.com/reference/iostream/manipulators/setw/
http://www.cplusplus.com/reference/iostream/manipulators/showpoint/
http://www.cplusplus.com/reference/iostream/manipulators/fixed/

You will probably want to make use of ***setw( )***, ***showpoint***, ***fixed***, ***precision( )***, and ***setf( )***

Extra Credit (1 Point):

In addition to the character count, generate another similar table that counts the words (anything separated by whitespace on both sides) in the sentence instead of the characters.

Goals:

The main goals for this assignment is for you to become familiar with the C++ STL map object and to learn how to format text output using C++ iostream manipulators and functions.

Deliverables:

Zip your program code & Makefile, rename the zip file to your first name underscore last name, and submit it on D2L in the Homework 3 dropbox.

Hints:

- Remember to test your program!
- When working on formatting the table output, try little examples to format text in various ways to get an understanding of what everything does.
- Do not try to tackle everything at once. Instead, decide on small chunks to change within the program and get those working first. For instance you could get all of the map data printed out without it being formatted, such as for the word **kitty** in the example below.

Character Count Percent
---

i 1 20%
k 1 20%
t 2 40%
y 1 20%

Additional Program Execution Examples:

```
Please enter a some text and press enter: happy, happy, joy, joy!!

Total letters in the user text = 24

Char      Count     Percent
---------------------------------
<SPACE>   3         12.50 %
!         2          8.33 %
,         3         12.50 %
a         2          8.33 %
h         2          8.33 %
j         2          8.33 %
o         2          8.33 %
p         4         16.67 %
y         4         16.67 %

Would you like to enter more text (y/n): n
Jeffreys-MacBook-Pro:hmwk3 jeffrey$
```

```
Please enter a some text and press enter: ^^^--@-@--^^^

Total letters in the user text = 13

Char      Count     Percent
---------------------------------
-         5         38.46 %
@         2         15.38 %
^         6         46.15 %

Would you like to enter more text (y/n): y

Please enter a some text and press enter: 18 +  2       = "20"

Total letters in the user text = 13

Char      Count     Percent
---------------------------------
<TAB>     2         15.38 %
<SPACE>   2         15.38 %
"         2         15.38 %
+         1          7.69 %
0         1          7.69 %
1         1          7.69 %
2         2         15.38 %
8         1          7.69 %
=         1          7.69 %

Would you like to enter more text (y/n): n
```