wordStatsPlus

Language/Type: C++ basics streams file input
Related Links: string istream

Write a function named **wordStatsPlus** that accepts as its parameter a string holding a file name, opens that file and reads its contents as a sequence of words, and produces a particular group of statistics about the input. You should report: the total number of lines; total number of words; the number of unique letters used from A-Z, case-insensitively, and its percentage of the 26-letter alphabet; the average number of words per line (as an un-rounded real number); and the average word length (also un-rounded). For example, suppose the file `tobe.txt` contains the following text:

```
To be  or not  TO BE,
 THAT  IS

really   the  question.
```

For the purposes of this problem, we will use whitespace to separate words. That means that some words include punctuation, as in "be,". For the input above, the call of `wordStatsPlus("tobe.txt");` should produce exactly the following output. The number of "unique letters" is 14 because the file contains 14 distinct letters of the alphabet from A-Z: a, b, e, h, i, l, n, o, q, r, s, t, u, and y.

```
Total lines = 4
Total words = 11
Total unique letters = 14 (53% of alphabet)
Average words/line = 2.75
Average word length = 3.45455
```

If the input file does not exist or is not readable, your function should instead print the following output:

```
Error, bad input file.
```

Constraints: Your solution should read the file only once, not make multiple passes over the file data.

. . .

```
void wordStatsPlus(string filename){

    ifstream istrm( filename );
    string line;

    char charInWord;   //uniq letters
    string unique = "";

    double totalwords = 0; //words per line & length
    int totalletters = 0;  //word length

    int j = 0; //ttl lines
    int lines = 0;
    if ( !istrm.is_open() ) {
            cout << "Error, bad input file." << endl;
    }
    else
    {
        bool inwrd = false;
        string prevline = " ";
        for (j = 0; getline( istrm, line); j++ ) {
            for(int i = 0;i<line.length();i++){
                if(isalpha(line[i])){
                    inwrd = true;
                    totalletters++;

                    charInWord = line[i];
                    charInWord = toupper( charInWord );
                    if (unique.find(charInWord) == std::string::npos){
                            unique += charInWord;

                    }

                }
                else
                {
                  if(inwrd || i == line.length())
                  {
                      totalwords++;
                      inwrd = false;
                  }
                }

            }
```

```
45          }
46
47          double roundunique = unique.length()/26.0;
48          roundunique *= 100;
49          double avgwordlines = totalwords/j;
50          double avglength = totalletters/totalwords;
51          if(totalwords <=1){
52              avgwordlines = 1;
53              avglength = totalletters;
54          }
55          cout << totalletters << endl;
56          cout << "Total lines = " << (j) << endl;
57          cout << "Total words = " << totalwords << endl;
58          cout << "Unique letters" << " = " << unique.size() <<
59               " (" << (int)roundunique << "% of alphabet)" <<endl;
60          cout << "Average words/line = " << avgwordlines << endl;
61          cout << "Average word length" << " = " << avglength << endl;
62      }
63      istrm.close();
64  }
```

Function: Write a C++ function as described, not a complete program.

Submit

You passed 2 of 5 tests. Try again.

test #1: `wordStatsPlus("wordStatsPlus-test1-data.txt");`

file input: wordStatsPlus-test1-data.txt:

```
To be  or not  TO BE,
 THAT  IS

really   the  question.
```

expected output:

```
Total lines = 4
Total words = 11
Total unique letters = 14 (53% of alphabet)
Average words/line = 2.75
Average word length = 3.45455
```

your output:

```
36
Total lines = 4
Total words = = 11
Unique letters = 14 (53% of alphabet)
Average words/line = 2.75
Average word length = 3.27273
```

differences:

```
0a1
> 36
2,3c3,4
< Total·words·=·11
```

```
< Total unique letters = 14 (53% of alphabet)
> Total words = = 11
> Unique letters = 14 (53% of alphabet)
5c6
< Average word length = 3.45455
> Average word length = 3.27273
```

result: fail
details: incorrect console output

test #2: wordStatsPlus("wordStatsPlus-test2-data.txt");
file input: wordStatsPlus-test2-data.txt:

```
Teenage Mutant Ninja Turtles,
Teenage Mutant Ninja Turtles,
Teenage Mutant Ninja Turtles,
Heroes in a half shell: turtle power!
END
```

expected output:

```
Total lines = 5
Total words = 20
Total unique letters = 18 (69% of alphabet)
Average words/line = 4
Average word length = 5.6
```

your output:

```
107
Total lines = 5
Total words = = 19
Unique letters = 18 (69% of alphabet)
Average words/line = 3.8
Average word length = 5.63158
```

differences:

```
0a1
> 107
2,5c3,6
< Total words = 20
< Total unique letters = 18 (69% of alphabet)
< Average words/line = 4
< Average word length = 5.6
> Total words = = 19
> Unique letters = 18 (69% of alphabet)
> Average words/line = 3.8
> Average word length = 5.63158
```

result: fail
details: incorrect console output

test #3: wordStatsPlus("wordStatsPlus-test3-data.txt");
file input: wordStatsPlus-test3-data.txt:

```
banana
```

expected output:

```
Total lines = 1
Total words = 1
Total unique letters = 3 (11% of alphabet)
Average words/line = 1
Average word length = 6
```

your output:

```
6
Total lines = 1
Total words = = 0
Unique letters = 3 (11% of alphabet)
```

```
Average words/line = 1
Average word length = 6
```

differences:

```
0a1
> 6
2,3c3,4
< Total words = 1
< Total unique letters = 3 (11% of alphabet)
> Total words = = 0
> Unique letters = 3 (11% of alphabet)
```

result: fail
details: incorrect console output

test #4: `wordStatsPlus("wordStatsPlus-test4-data.txt");`
console output:

```
Error, bad input file.
```

result: pass

test #5: `wordStatsPlus("bogus-data.txt");`
console output:

```
Error, bad input file.
```

result: pass

Need help?

Stuck on an exercise? Contact your TA or instructor.

If something seems wrong with our site, please contact us.