COSC2430: Data Structures and Algorithms
HW0: Counting words and numbers in a text file

1 Introduction

You will write a C++ program to count words and numbers in a plain text file. Words and numbers can be repeated. You can use any algorithm and data structures that you prefer, as long as the results are correct. It is preferred, but not necessary, that your algorithm is as efficient as possible, both in processing time as well as memory management.

2 Input Specification

The input is one text file, with words and integers, separated by any other symbols including spaces, commas, period, parentheses and carriage returns. Keep in mind there can be be multiple separators together (e.g. many spaces, many commas together).

The input is simple. You can assume a word is a string of letters (upper and lower case) and a number a string of digits (an integer without sign). Words and numbers will be separated by at least one non-letter or one non-digit symbol.

Length: You can assume one word will be at most 30 characters long and a number will have at most 10 digits.

Repeated strings: words and numbers can be repeated. However, you are not asked count distinct words or compute frequency per word, which require algorithms and data structures to be covered in the course. Therefore, you just simply need to count word or numver occurrences.

For this homework we will not ask you to consider all potential inputs. Notice you are not asked to handle floating point numbers, which require decimal point and scientific notation. Therefore, a period can be considered a separator. Notice also we will avoid abnormal strings like variable names combining letter and digits (X1, student2, ..).

Example of input files (between the lines) and result (on the screen):

```
input1.txt
------------------------------------------------------------
The cat is [there]
10   20 3.1416,,1000
another cat
------------------------------------------------------------

Program output:
words=6 numbers=5

input2.txt
------------------------------------------------------------
Intel CPU, Western Digital Hard disk
4 cores, 16 GB RAM,, ...
$
------------------------------------------------------------

Program output:
words=9 numbers=2
```

Notice words and numbers can appear in any order, one or more per line. Also, notice there may be extra symbols (separators) you should simply skip since they are neither words nor numbers.

3 Program and input specification

The main program should be called "count". Call syntax is as follows (from the OS prompt):

```
count filename=input1.txt
```

Notice that the file name will not necessarily be the same every time. Therefore, your program will have to take that into account.

The output should be sent to the screen, as shown in the examples. Use only this format. Avoid introducing extra symbols or strings since your program will be graded automatically. For instance, avoid using phrases like "the output is", or "the word count is").

```
words=4 numbers=5
```

4 Requirements

- Homework is **individual**. Your homework will be automatically screened for code plagiarism against code from the other students and code from external sources. If you copy/download source code from the Internet or a book it is better you acknowledge it in your comments, instead of the TAs detecting it. Code that is detected to be copied from another student (for instance, renaming variables, changing for and while loops, changing indentation, etc) will result in "Fail" in the course and being reported to UH upper administration.
- You can develop your program on any C++ compiler (MS Visual C++, Borland C++, Intel C++), BUT you must test your program in GNU C++. The TAs have no obligation to test your program on any compiler other than GNU C++.
- Once you have some experience with the compiler start your assignment. Remember that all the following homework sets will build upon each other, so it is important that you start with the right foot. Try to write your classes and procedures in a way that can be used in the future and
- **don't forget to comment your code**, especially complicated code. You must include a short summary of your main functions in the main cpp file.
- Your program should write error messages to the screen with invalid input. Your program should not crash, halt unexpectedly or produce unhandled exceptions.
- Be prepared for input going from very simple to more complicated. Consider empty input, zeroes and inconsistent information. Each crash exception will be -20.
- Test cases. Your program will be tested with 10 test cases, going from easy to difficult. You can assume 80% of test cases will be clean, valid input files. If your program fails an easy test case 10-20 points will be deducted. A medium difficulty test case is 10 points off. Difficult cases with specific input issues or complex algorithmic aspects are worth 5 points.
- A program not submitted by the deadline is zero points. A non-working program (compiles, but does not run with any test case) is worth 10 points. A program that does some computations correctly, but fails several test cases (especially the easy ones) is worth 50 points. Only programs that work correctly with most input files that produce correct results will get a score of 80 or higher.
- Correctness is more important than speed. You should always err on the side of caution submitting a slow program that works correctly than a fast program that fails in many cases.
- Since this homework will only be "Pass/Fail" if your program scores more than 50 points, you will pass.
- Homeworks will gradually increase in level of difficulty. It is recommended that if a program gets "Fail" then the student drops the course and talks to the undergraduate advisor to take a remedial programming course.