CS 2022 – Fall 2009
Assignment #2
9/14/2009
Due: Friday 9/18/2009 11:59 PM

In this assignment you are asked to write a small C program to demonstrate command of principles we have discussed in class so far.

Characters Frequency Calculator

You are asked to write a C program `charfreq.c` that will take-in **multiple** command line arguments and output statistical information about these arguments. More specifically, your program will output a table listing all the characters (letters, numbers, spaces, or whatever .. i.e. you do not have to check if the contents of the passed-in strings are alphabet letters).

Here are some example runs:

- ```
  :~> ./charfreq "HELLO CLASS"
  H-1
  E-1
  L-3
  O-1
   -1
  C-1
  A-1
  S-2
  ```
- ```
  :~> ./charfreq i can count 123
  i-1
  c-2
  a-1
  n-2
  o-1
  u-1
  t-1
  1-1
  2-1
  3-1
  ```

Notice that each character is printed in the table **exactly once**. Also notice that the space character was printed in the first example but not the second because the first example had only one command line argument which happened to contain the space (that is why the quotes were used), while in the second example, three separate command line arguments were passed-in and no spaces were in any of them. **The characters must be printed in the order as they appear in the arguments list**.

Technicalities

In this assignment you have to demonstrate your knowledge of "complex" C types. So you **must** store each character count in a instance of a `struct` you define, and you have to link `structs` to one

another via pointers.

So for example, you should define and use something like this:

```
struct char_count {
     char c;
     int count;
     struct char_count *next;
};
```

Of course you may use `typedef` to make dealing with that `struct` a bit less wordy/ugly. The `next` pointer points to the next character count, and the pointer in the last `struct` instance should point to `NULL`. Thus, printing out the table should just be a matter of starting at the first `struct`, printing that character and its its count, then going down to the next character count `struct`.

Submission and Testing

Submit your work on CMS (http://cms.csuglab.cornell.edu/) by the deadline. Make sure you have been added to CMS early on and otherwise contact me via email. You should submit the **source code** (the .c files) of your application and **not** the compiled binaries.

Your programs will be compiled with `gcc` and tested on a linux environment. You are free to choose the environment of your liking to develop your solutions, but keep in mind that testing will be on a fixed environment, and your application is expected to run on that.

Academic Integrity Reminder

Remember that you may have general discussions about how to approach this problem with your peers, but you should work on the final solution by yourself alone. If you are stuck or are having trouble, you may email me or talk to me after class on Monday or during my office hour on Wednesday.

Good Luck!