CSCI 2500 — Computer Organization
Lab 03 (document version 1.0)

- This lab is due by the end of your lab session on Wednesday, September 19, 2018.
- This lab is to be completed **individually**. Do not share your code with anyone else.
- You **must** show your code and your solutions to a TA or mentor to receive credit for each checkpoint.
- Labs are available by 6:00PM on Mondays before your lab sessions. Plan to start each lab early and ask questions during office hours, in the discussion forum on Submitty, and during your lab session.

1. **Checkpoint 1:** Download the `lab03-data.dat` file and attempt to decipher what the file contains. From the shell, try viewing the file in a text editor. Also try using `cat` and `hexdump` to learn what you can about the file.

   Any guesses as to what data is stored in this file?

   Don't read any further until you try to decipher what's in this file!

   Next, determine the size (in bytes) of the `lab03-data.dat` file. To do so, you can use the `ls` command. If you do not already know the flag to use with `ls`, check out the `man` page for `ls`.

   Assume that the `lab03-data.dat` file contains an array of `int` variables. Write a C program to read this data file entirely into memory, then display the data using a loop.

   To accomplish this, you must call each of the following functions exactly once: `calloc()`, `fopen()`, `fread()`, and `fclose()`. In other words, allocate enough memory for this file (you can hard-code the number of bytes needed), then use only one call to `fread()` to read in exactly that number of bytes.

   Finally, display the data as follows:

```
Data point #  0: <int-value>
Data point #  1: <int-value>
Data point #  2: <int-value>
...
Data point #183: <int-value>
Data point #184: <int-value>
Data point #185: <int-value>
```

2. **Checkpoint 2:** Modify your solution for Checkpoint 1 by instead assuming that the given `lab03-data.dat` file contains an array of `unsigned long` variables.

   Display the data as follows:

   ```
   Data point # 0: <unsigned-long-value>
   Data point # 1: <unsigned-long-value>
   Data point # 2: <unsigned-long-value>
   ...
   Data point #90: <unsigned-long-value>
   Data point #91: <unsigned-long-value>
   Data point #92: <unsigned-long-value>
   ```

   Did you figure out what this data file contains yet?

   Note that we can interpret the raw data however we like, treating it as an array of `int` values, an array of `unsigned long` values, a string of printable characters, etc. Only one interpretation turns out to be correct, though.

3. **Checkpoint 3:** For the third and final checkpoint, write a C program to read a text file into memory one line at a time. Use `fgets()` to accomplish this. And assume a line is no more than 128 bytes long, meaning you can declare a character array buffer as `char line[128]`.

   Next, without using square bracket `[]` notation or any `int` (or other numeric) variables whatsoever, traverse each line and display each alphanumeric character you find. To accomplish this, use pointer arithmetic and the `isalnum()` function in `ctype.h`.

   At the end of each line, display the total number of alphanumeric characters found on that line. Use the format shown in the example below. Again, do not use any `int` or other numeric variables to accomplish this.

   Finally, if no alphanumeric characters are found on a given line, do **not** display anything, not even a blank line.

   To test your code, start with small test cases, then download the `book-1984.txt` file and test with that as input. Your output in this case should start as follows:

   ```
   PARTONE [7 alnum chars]
   Chapter1 [8 alnum chars]
   ItwasabrightcolddayinAprilandtheclockswerestrikingthirteen [58 alnum chars]
   WinstonSmithhischinnuzzledintohisbreastinaneffortoescapethe [60 alnum chars]
   vilewindslippedquicklythroughtheglassdoorsofVictoryMansions [59 alnum chars]
   thoughnotquicklyenoughtopreventaswirlofgrittydustfromentering [61 alnum chars]
   alongwithhim [12 alnum chars]
   ...
   ```

   HINT: Use pointer arithmetic to achieve this checkpoint.