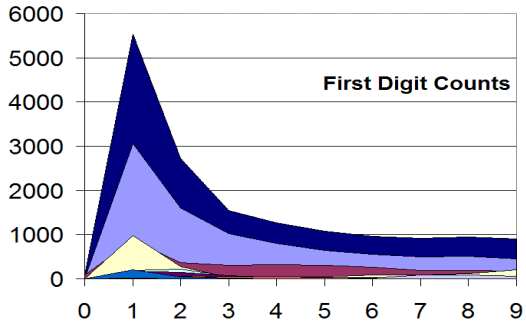
CS 218 – Assignment #11, Part A

Purpose: Become more familiar with operating system interaction, file input/output operations, and file I/O buffering.
Due: Monday (7/01)
Points: 225 (grading will include functionality, documentation, and coding style)

Description:

Benford's Law[1] describes a surprising pattern in the frequency of occurrence of the digits 1–9 as the first digits of natural data. You might expect each digit to occur with equal frequency in arbitrary data. In truly random data (over appropriate ranges) each digit does appear with equal frequency. However, a substantial amount of natural data from diverse sources do not exhibit a uniform distribution. Instead, 1 is more common than 2, 2 more common than 3, and so forth. We will test this law using a program to read data source files and count the leading digits.



Assignment:

Write an assembly language program that will read a data source file, count the leading digits, and output a formatted text-based graph of the results. The graph will be written to the file and, optionally displayed to the screen. The program should read the screen display option (true or false) and both file names (input and output) from the command line. An example command line is:

```
ed-vm% ./benford  -i books.txt  -o results.txt  -d F
```

The input file should be specified as "-i <filename>" and the output file must be specified as "-o <filename>". The display option should be specified as "-d <T/F>". The program must perform error checking. If no command line input was entered, invalid/incorrect file names were entered, or no files matching the specification were entered an appropriate error message should be generated. If an error occurs, the appropriate error message should be printed and the program terminated.

The input files will be specially formatted. The first 5 lines will be header information, and the first 7 characters of each line will be title characters (that must be skipped). Each number will be of different size. You need only check for the first digit. You do not have to handle files that do not conform. You will be provided a series of example files for testing. Additionally, the output graph should follow the provide example (with the octal counts). The number of *'s to print is determined by dividing the count by SCALE. If the count is 0, display no stars. If the total count is < 100,000, the scale is 100. If the total count is ≥ 100,000 and < 500,000 the scale should be set to 1000. If the total count is ≥ 500,000 and < 1,000,000 the scale should be 2500. If the total count is ≥ 1,000,000 and < 2,000,000 the scale should be 6000. If the total count is ≥ 2,000,000 the scale should be 12000.

To ensure overall efficiency, the program **must** perform buffered input and output with a buffer size of BUFF_SIZE (originally set to 500,000). Note, this will be changed for part B. You must use the provided main routine. ***The main program must not be changed in any way.*** All your procedures must be in a separate, independently assembled source file.

1 For more information, refer to: http://en.wikipedia.org/wiki/Benford's_law

Submission:
When complete, submit:

- A copy of the **source file** via the class web page (assignment submission link) by class time. ***Assignments received after the due date/time will not be accepted.***

Testing
A batch file to execute the program on a series of pre-defined inputs will be provided. *Note*, please follow the I/O examples. The test utility should be downloaded into an empty directory and the program executable placed in that directory. The test script can be executed as follows:

```
ed-vm$ ./a11tst benford
```

The test script compares the program output to pre-defined expected output (based on the example I/O).

Independent Assembly
Note, your procedures must be placed into a *separate file* and linked with the provided main. Refer to the handout for directions how to link multiple files. Only the procedures file, not the provided main, will be submitted on-line. ***As such, you must not change the provided main!***

Example Executions:
The following execution will read file "books.txt" and create a file named "results.txt".

```
ed-vm% ./benford -i books.txt -o results.txt -d F
ed-vm%
```

The output file, "results.txt" contains the following lines:

```
ed-vm% more results.txt

CS 218 Benfords Law
Test Results

Total Data Points: 21662

  0  0        |
  1  5760     |******************************
  2  3106     |****************
  3  1772     |**********
  4  1441     |********
  5  1200     |******
  6  1060     |*****
  7  766      |*****
  8  767      |*****
  9  704      |****
               ---------------------------------------------

ed-vm%
```

Note, if the display option (-d) was set to T, the above graph would also be displayed to the screen.