Assignment #7 – Buffered I/O **CS 218 Fall 2021**

Purpose: This program will involve using the buffered I/O algorithm to efficiently analyze a large text file.

You will need to write this program from scratch.

Program Specifications (Part A)
This program will read in a text file and calculate the total number of words and average word size of the file.

The program will need to take in the name of a file as a command line argument. An optional third argument "-echo" may also be included. Check for invalid arguments.

Attempt to open the indicated file outputting an error and end the program if the file does not open successfully.

Read through the file using the functions described below, then print out the word count and average word length to the console. If -echo is included, print each character to the console as it is read.

Required Functions

getWordCountAndAverage
This function should take two arguments by reference, the number of words found in the file and a calculated average word length. A word is any sequence of characters that does not contain a whitespace character (space, tab, linefeed, carriage return, etc). Multiple white spaces may separate words. Keep a count of all letters for the average word size calculation. Use an integer (round down) average.

getCharacter
This function should retrieve a single character from the buffer as described in the buffer I/O algorithm. Store the character into a pass by reference argument. The buffer variables and file descriptor may be accessed globally. Return 1 if a character was retrieved succesfully, 0 if there are no more characters to retrieve, and -1 if an error occurred during the read system service call.

Use a buffer size of 100,000 characters.

Additional assembly functions (to print out a decimal value for example) may also be included.

In Main

Make sure to use main: instead of _start: as your starting function and use **g++ -g -no-pie assemblyFile.o** to link your program.

Write Up (Part B)

In a document file (.docx or .odt), experiment with the buffer size by running the program with buffer sizes of 1, 1000, and 100,000.

For each buffer size, run the program using the Linux timeprogram on the combined.txt file. It may take several minutes to run.

```
time a.out combined.txt
```

Create a table using the real, user, and system times and calculate the percentage of time each buffer size spends in system time(sys).

(sys time)/(real time)*100%

Explain why certain buffer sizes finish faster than others in 200 words or less (preferably much less).

Submission

Once you are satisfied with the program, upload the assembly source code (.asm) file to the class website.

Example Execution

```
$ ./a.out test.txt -echo
File Text:
Testing, testing.
Hello, World!

Word Count: 4
Average Word Length: 6
```

Example Execution

```
$ ./a.out test.txt -ech
Invalid argument.
```

Example Execution

```
$ ./a.out test.txt -echo 123
Incorrect number of arguments.
```

Example Execution

```
$ ./a.out
To use this program include the name of the file you wish to analyze.
-echo may be added to print the file to the terminal.
```

Example Execution

```
$ ./a.out fake.txt -echo
Could not open “fake.txt”.
```