Christine Do, Jessica Guo
CS214 Assignment02 Threads and Processes due November 22, 2016 @ 11:55PM

Compression Algorithm

If the number of files requested does not evenly divide into the length of the string, the remaining leftover length is added to the first file. For example, a string of length 11 divided into three files would result in the strings to be compressed into 5 3 3.

Numerical characters result in a warning message printed out; they are skipped.
Whitespaces are skipped (by Professor Francisco's spec).
Other invalid characters are skipped and compression continues.

Expected input is txt file.
Output is in the form of .txt files.

Program will exit if input file already has compressed files in directory (it will not compress test.txt if test_LOLS.txt exists).

Program will exit if number of files called for is greater than the length of the string. (Ie, string length = 4, user calls for 6 files).

Up to and including 100 processes/threads can be made. Other test cases are included in testplan.txt

Threaded Implementation

compressT_LOLs.c

To call the threaded implementation of LOLS, it is assumed that the user is working in terminal. Compile compressT_LOLS.c

```
gcc -o compressT_LOLS compressT_LOLS.c
```

Input format should be filename, # of output files wanted.

Ex: `./compressT_LOLS filename.txt 3`

Process Implementation

compressR_LOLS.c, compressR_worker_LOLS.c

To call the process implementation of LOLS, it is assumed that the user is working in terminal.
Compile both compressR_LOLS.c and compressR_worker_LOLS.c

```
gcc -o compressR_LOLS compressR_LOLS.c
gcc -o compressR_worker_LOLS compress_worker_LOLS.c
```

Input format should be filename, # of output files wanted. Only compressR_LOL is needed.
CompressR_LOL will check that arguments are valid and that the file has not already been compressed. It will also check that the file can be open. After doing so, it will execute compressR_worker_LOLS, which will begin forking child processes.

Ex: `./compressR_LOL filename.txt 4`