Shoulders.c design

I structured my code into a main function and a file processing function. It could have been done into a single main function but I decided to split it up as it was easier when implementing special input cases such as “-” or no file arguments.

At a high level my shoulder implementation reads in 4096 bytes of data from the text input into a buffer . It then loops through this buffer and searches for a newline character. When it finds the newline character it writes everything to stdout up to and including the newline. If it gets to the end of the buffer and there are more lines to print, it will flush the rest of the buffer to stdout and read in more data which will overwrite the buffer. This process repeats until the desired number of lines is written.

main()

Get the number of lines to print from command arguments

Process any invalid arguments, use warn(3) or fprintf(stderr)

Loop through the file arguments
Call procss_file() on each file
If the file is ‘-’
Use stdin
Close file

Process_file(fd, lines)
Allocate a static buffer, could be anything but the bigger the better the performance

Loop until desired number of lines
Read in bytes to buffer
Is no more bytes to read
Break out of loop

Have a counter to keep track of where the last newline was in array | start = 0
Loop through the buffer array
If there is a newline
Write to stdout starting from the buffer[start] to the newline
Increment line printed

Replace start with the counter to mark where last newline was

If all the desired lines are printed
Break

If there are more lines to be printed then write out the rest of the buffer because it is going to be overwritten