# Assignment 5 – Buffered I/O

## Description:

This assignment is to write a C program that loads files into a file control block array to be read and copied into a caller's buffer. When files are loaded from the file control block, they are loaded into the fcb buffer one 512 block at a time. Once loaded, the caller's buffer is filled from the caller's request.

# Approach / What I Did:

I have programmed this assignment multiple times with different approaches. First, I did some basic code to just load the file into a single block to be read with print statements to see the behavior of the different functions provided. Unfortunately, we were not provided full specifications for the function that this assignment hinges upon: LBAread, so quite a bit of testing was necessary to figure out what it returned and how it behaved with file reads. For example, it was unclear as to whether it would return only the number of bytes remaining in a file or if the file byte count needed to be tracked from our end.

I then diagrammed out the logic necessary to handle the fcb array, the reads, and the buffer. Once I completed that, I turned to starting the coding. As I coded, I revised and revisited my diagrams and notes as I ran into new hurdles. I also visited office hours and found some new approaches to the assignment.

Logically speaking, if the requested data count exceeds the amount left in the file, we should only return what is left in the file. After this, there are two cases: the requested amount is less than or equal to the amount left in the fcb buffer or greater than what is left in the buffer. If less, we simply copy over the requested amount and increment the buffer offset. If greater, we need to copy over whatever is left in the buffer and handle what is left in the request. This is done by calculating how many blocks the request needs and then copying over full blocks until only a partial block is left (if there is a partial block). Then lastly, whatever is left is copied over.

In the end, I scrapped my code multiple times to recode the assignment from scratch to try to solve some issues. I have two versions of the assignment in two different branches (the main branch is my submission) which behave almost identically (the other version is in the "test" branch).

During all this time, I did some pair-programming with another student in the class to assist in reasoning out anything that I may have forgotten.

#### **Issues and Resolutions:**

This is not really an issue since this happened during my exploratory stage for the assignment, but in the beginning, no data was being copied over.

As I continued progressing, I ran into the issue of the program not reading and copying over all the data. This was resolved by writing out all the cases that I needed to handle. After writing all the cases, I was able to reason out the cases that I forgot to include. After this, the code pretty much wrote itself.

Even after I resolved most of the issues, I scrapped all my code and tried to reimplement the program in a different way (again in the "test" branch).

Analysis: None required

# Screen shot of compilation:

```
parallels@ubuntu-linux-22-04-desktop: ~/Rep... Q = - - ×

parallels@ubuntu-linux-22-04-desktop: ~/Repos/csc415-5$ make gcc -c -o b_io.o b_io.c -g -I. gcc -o Kim_Mark_HW5_main b_io.o buffer-mainM1.o -g -I. parallels@ubuntu-linux-22-04-desktop: ~/Repos/csc415-5$
```

SCREENSHOTS OF EXECUTION ON SUBSEQUENT PAGES

## Screen shot(s) of the execution of the program:











