# Assignment 5 – Simple Shell with Pipes

## **Description:**

This assignment was designed to help us prepare for our file system group project by practicing block operations, managing memory, tracking the information of multiple files, and generally understanding the functionality of low level files.

This program implements a simple file system interface in C using similar open, read, and close functions to Unix systems.

## Approach:

I'll be breaking this assignment into two parts: opening and closing the file (b\_open and b\_close), and then reading the file (b\_read).

#### b open & b close

To open the file, we need to fill up the File Control Board which is a struct that contains the data of the file, and return the file descriptor which is essentially the index of the file's FCB within the FCB array. We first check if the FCB array has been initialized and if there is an available FCB. We can call get\_FCB() which will return an available FCB if it exists, and we break out if none are found i.e. they are all in use. We can then call GetFileInfo() to retrieve the data for that file. If the file data returns as null, the file doesn't exist and we break out. Otherwise I put the file info into the FCB for the file.

I then added some more fields to the FCB struct that will be needed for the rest of the assignment: a mediary buffer to read the file, an index of where we are in the buffer, and an index of where we are in the file.

I then went back to b\_read to allocate memory for the buffer and set the file and buffer index to 0 since it has just been opened. If the memory allocation fails, then break out. Otherwise, return the file descriptor.

To close the file we first check if the file descriptor is valid and the file exists. If it does, we simply free the memory allocated for the buffer and set the fi field to null to indicate that it is free for use.

### <u>b\_read</u>

To read the file, we can break it into three steps:

- 1. If there is any data in the buffer, copy it into the caller buffer
- 2. If the remaining amount to be read is larger than one complete block, read the complete blocks using LBAread directly to the caller buffer

3. If we have any remaining bytes to be read less than a complete block, LBAread() in the final block to our buffer and copy the remaining amount of bytes requested to be read to the caller buffer

We first check that the system has been initialized, the file descriptor is valid, and that the FCB has valid file data. If any of these checks fail, we break out. We then initialize a pointer to access the fields of the FCB in the FCBarray, and a counter to track how many bytes have been read.

If the amount of bytes requested to be read is more than the remaining bytes left in the file, we set the remaining space to the requested amount so as to not go past the EOF. If the requested amount of bytes is 0, then we return 0 to indicate that no bytes were read.

We can now begin the reading process by checking if the buffer offset is greater than 0 which would indicate that there is data currently in the buffer. If the requested amount of bytes to read is more than the remaining space in the buffer, then we will copy over just enough to fill the buffer. We use memcpy() to copy the data into the caller buffer and update all the counters and offsets.

We then calculate if the remaining amount of bytes requested to read is larger than one complete block. If it is, we can copy an entire block/blocks directly to the caller buffer. We call LBAread() directly to the caller buffer with the amount of blocks that can be read and update the counters and offsets.

Lastly, if there are still bytes requested to be read, we do another LBARead call to our buffer with the last block of the requested amount of bytes, use mcmcpy() to copy over just the amount of bytes requested to the caller buffer, and increment the offsets and counters to ensure that the next b\_read() call will start at the net available byte in the buffer. We then return the amount of bytes read.

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

The first issue I had was that my output seemed to be skipping or rereading some portions of the data file. I wasn't sure where my error was and began drawing out my logic on paper. I realized that after reading complete blocks directly into the caller buffer I wasn't resetting my fileOffset correctly. I was setting it equal to the amount of bytes that I had just read in rather than adding it in case some bytes had already been read in the previous step. This was making the program think it was further back in the file and wasn't accounting for the bytes it had already read.

My second issue was very similar in that my output still appeared to be missing sections of the text within DATA. I knew my issue was either in my adjustments of my offsets and counters, or the starting position in the buffer that I was handing to LBARead() or memcpy(). With more drawing out my logic and print statements to debug I found that I wasn't handing the correct

starting position to my LBAread() call to read multiple blocks. If we had already read 30 blocks into the buffer in the previous step, these were being overwritten because I wasn't adding the bytes already read to the caller buffer's position. After adding these to the starting position of the destination, my output looked correct.

## **Screenshot of compilation:**

```
student@student:~/Documents/CSC415/csc415-assignment-5-buffered-io-smeerj$ make
gcc -c -o b_io.o b_io.c -g -I.
gcc -o Bhagat_Arjun_HW5_main b_io.o buffer-main.o -g -I.
student@student:~/Documents/CSC415/csc415-assignment-5-buffered-io-smeerj$
```

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

```
student@student:~/Documents/CSC415/csc415-assignment-5-buffered-io-smeerj$ make run
gcc -c -o b_io.o b_io.c -g -I.
gcc -o Bhagat_Arjun_HW5_main b_io.o buffer-main.o -g -I.
./Bhagat_Arjun_HW5_main_DATA_DecOfInd.txt_CommonSense.txt
The unanimous Declaration of the thirteen united States of Am
Perhaps the sentiments contained in the following pa
erica, When in the Course of human events, it becomes necessary for one people to
ges, are not yet
sufficiently fashionable to procure
dissolve the political bands which have conn
them general favor; a long habit
of not thinking a thing wrong, gives it a super
ected them with another, and to assume among the powers of the earth, the separate
ficial appearance of
being right, and raises at first a formidable outcry in
and equal station to which the Laws of Nature and of Nature_
defense of
custom. But the tumult soon subside
s God _ntitle t_em, a decent respect to the opinions
s. Time makes more converts than
reason.
of mankind requires that they should declare the causes which
As a long and violent abuse of power, is generally the Means o
impel them to the separation.
We hold these truths to be self-evident, that all men are created equal, that they a
re endowed by their Creator with certain unalienable Rights, that among these are Li
fe, Liberty and the pursuit of Happiness.--That to secure these rights, Governments
are instituted amona Men. derivina their just powers from the consent of the governe
```













