Brandon Prophete

CS340-Section 26

Project 1 Write-up and Explanation

When completing this project/program the goal was to make use of shared memory and the fork() system call command to read a source file in 128-character chunks into shared memory and then write said content into a target file via a producer parent process that will read the file, where the consumer child process will take over and write it to the file. A pipe() system call will also be implemented between the parent and child. One of the biggest challenges in testing out my program was figuring out how to complete many of the functions without using the C Standard Library, such as reading and writing to and from files. My knowledge of the C language was limited so it was quite the learning curve trying to circumvent this problem. I was able to solve these numerous issues through using the <fcntl.h> library for creating,reading, and writing to and from files and all file control functions. I also included all of the other libraries that went together with the producer and consumer like <sys/shm.h>, <sys/stat.h>, and <sys/mman.h>. Another library I used was <unistd.h> as it helped to fix an error I had with ftruncate() as when I tried testing my producer and consumer it kept giving me an error with the compiler trying to recommend I use strncat instead. These libraries I believe fall under POSIX system I/O calls and they greatly helped with errors and bug fixes. One major issue I'm having is finding a method for turning my integer into a character without using the sprintf() function call. The way my program works is I use pointers of the buffer, in, out to be part of the shared memory so I first deal with the parent process and try to read in shared memory into the file. I don't know if my process is correct, but I really tried to follow the notes and do the best that I can on it. This was the same for the child process, but I do think I figured out a way to stop both processes when they are empty. It was also kind of a challenge determining where I should put the pipe process with my child and parent process, but that issue soon resolved itself with some basic formatting. Going back to turning my integer into character I thought there was no other way to do it until I thought of a method to complete. First I would take the modulus of the int which is n % 10 to get the last digit and then add the character 0 (ascii values) to convert the digits to characters. Also since they would be printed in reverse order I used the system call strrev to get them back into the right order. Yet, still that was giving me an error so I decided it would be better to just leave it out. One of the last issues I'm having is using some system calls such as write() and pipe() giving me issues during compilation on the C compiler I was using, but I tried using a Linux virtual machine for C compilation and execution. Also you may notice that I don't have the didn't use the arguments in the code, that's because of how the compiler couldn't find my code so I used the sourceFile to get my information and my .exe has -lrt at the end due to shm_link. Even though the output needs a lot of formatting work and values too.