Jonathan Carreiro

Lab 7

Part 1:

a. The newly created processes do have their own copies of each variable. These copies are created when the process is created so the value of each variable for the child is based on its current value in the parent at the time of creation. For example, the variable "value" was at 60 when child 1 was created so it starts there. The same is true for child 2. However, child 1 and child 2 have separate versions of "value" so incrementing in C1 does not change it in C2. Hence why you see the same value showing up twice; once in C1 and once in C2.

b. In my program, child 2 is the first process to start printing but this is not a general rule. In general, there is no set order to which process will go first. Because there are no waits() implemented, the processes run concurrently. This means that the print statements are first come, first served. So, they could be in a different order each time the program is run.

c. Added in a sleep() after successful creation of process 2 to ensure that it does not print until process 1 is finished.

Part 2:

a. Both the child and the parent processes share their argv list.

b. Both the child and the parent process seem to share their envp list. I was not able to completely compare every last part, but they seemed the same after a brief look.

c. Both the child and the parent print to the same file as the file descriptor is shared between them.

Part 3:

a. Both the PID and the PPID are retained in the PCB after the exec.

b. The file created exists and it contains the PID/PPID of the original process as well as the PID/PPID of the newly created process.

c. The environment variables are inherited in after the exec, even after changing envp[1] to be my name.