**CIS 452 02 – Lab 2 – Gabrielle Munson & Gabe Baksa**

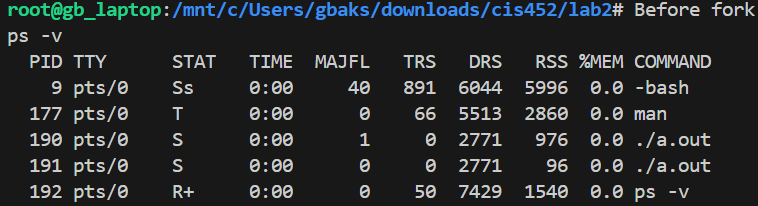
### **Sample Program 1**

1. **Compile and run Sample Program 1, how many lines are printed by the program?**

3 lines are printed

1. **Describe what is happening to produce the output observed.**

After the fork is created, the process is duplicated and that results in “after fork” being printed twice, giving us three lines printed.

1. **Insert a 10-second call to the function sleep() after the fork in Sample Program 1 and recompile. Run Program 1 in the background (use &). Consult the man pages for the ps (process status) utility; they will help you determine how to display and interpret the various types of information that is reported. Look especially for "verbose mode" or "long format". Then, using the appropriate options, observe and report the PIDs and the status (i.e., execution state info) of your executing program. Provide a brief explanation of your observations.**

PIDs: 190 and 191

STATs: S and S

Interruptible sleep (waiting for an event to complete)

Does not have the + because it is running in the background

### **Sample Program 2**

1. **Create a diagram illustrating how sampleProgramTwo executes (i.e., give a process hierarchy diagram (see figure 3.8 from the textbook)). Run the program several times with small input values (e.g., 2...5) to help you understand exactly what is happening.**



1. **In the context of process state, process operations, and especially process scheduling, describe what you observed and try to explain what is happening to produce the observed results.**

One process is first created in the background, when the fork() function is executed the first process is then duplicated and creates a child process. The fork() function is then come across again resulting in the original process and child process to both be duplicated. For process scheduling, the process that arrives at the CPU first is the one that will fork first, there is no priority of one over the other.

### **Sample Program 3**

1. **Provide the exact line of code that you inserted for the wait() system call (2pts).**

child = wait(&status);

1. **Which prints first, the child or the parent? Why? Describe the interaction between the exit() function and the wait() system call. You may want to experiment by changing the value to better understand the interaction (6 pts).**

The child prints first due to the wait() system call that ensures that the child terminates before continuing.

### **Sample Program 4**

1. **When is the second print line ("After the exec") printed? Explain your answer.**

The second print line (“After the exec”) will never be printed. If the program succeeds, it outputs (“Before the exec”), lists the output of the command, and terminates the program. If the program fails, it outputs (“Usage: must supply a command”) and terminates the program. These are the only two output cases for the program thus never printing (“After the exec”)

1. **Explain how the second argument passed to execvp() is used (3pts)?**

The second argument passed to execvp() is used to provide the program being executed with its command-line arguments.

### **Programming Assignment**

