**CSC4100/5100 – Homework 1 – Processes**

**https://pages.cs.wisc.edu/~remzi/OSTEP/cpu-api.pdf**

1. Write a program that calls fork(). Before calling fork(), have the main process access a variable (e.g., x) and set its value to something (e.g., 100). What value is the variable in the child process? What happens to the variable when both the child and parent change the value of x?
   1. The value remains the same in the parent process as is in the child process.
      1. Created before as x = 100 and still x = 100 in child process
   2. The value changed in the parent process will not be touched by the child process as once the child process is complete the value of x still is 100 even when the child process changes x to 50.
2. Write another program using fork(). The child process should print “hello”; the parent process should print “goodbye”. You should try to ensure that the child process always prints first; can you do this without calling wait() in the parent?
   1. Yes you can with the sleep() command.
3. Write a program that calls fork() and then calls some form of exec() to run the program /bin/ls. See if you can try all of the variants of exec(), including (on Linux) execl(), execle(), execlp(), execv(), execvp(), and execvpe(). Why do you think there are so many variants of the same basic call?
   1. Variety allows for developers to have a flexibility into what kind of input they would prefer whether it be a whole execution path or just a environment variable to find an executable
4. Write a program that creates two children, and connects the standard output of one to the standard input of the other, using the pipe() system call.