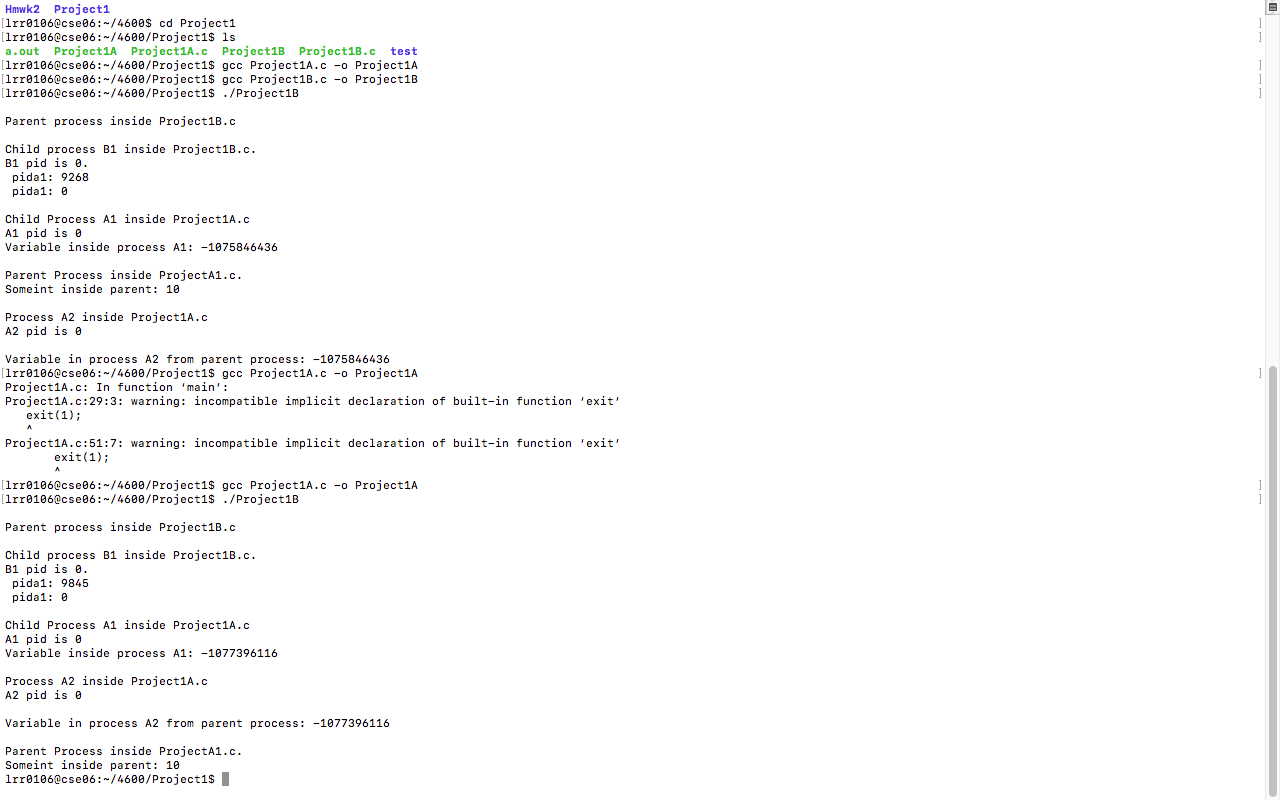
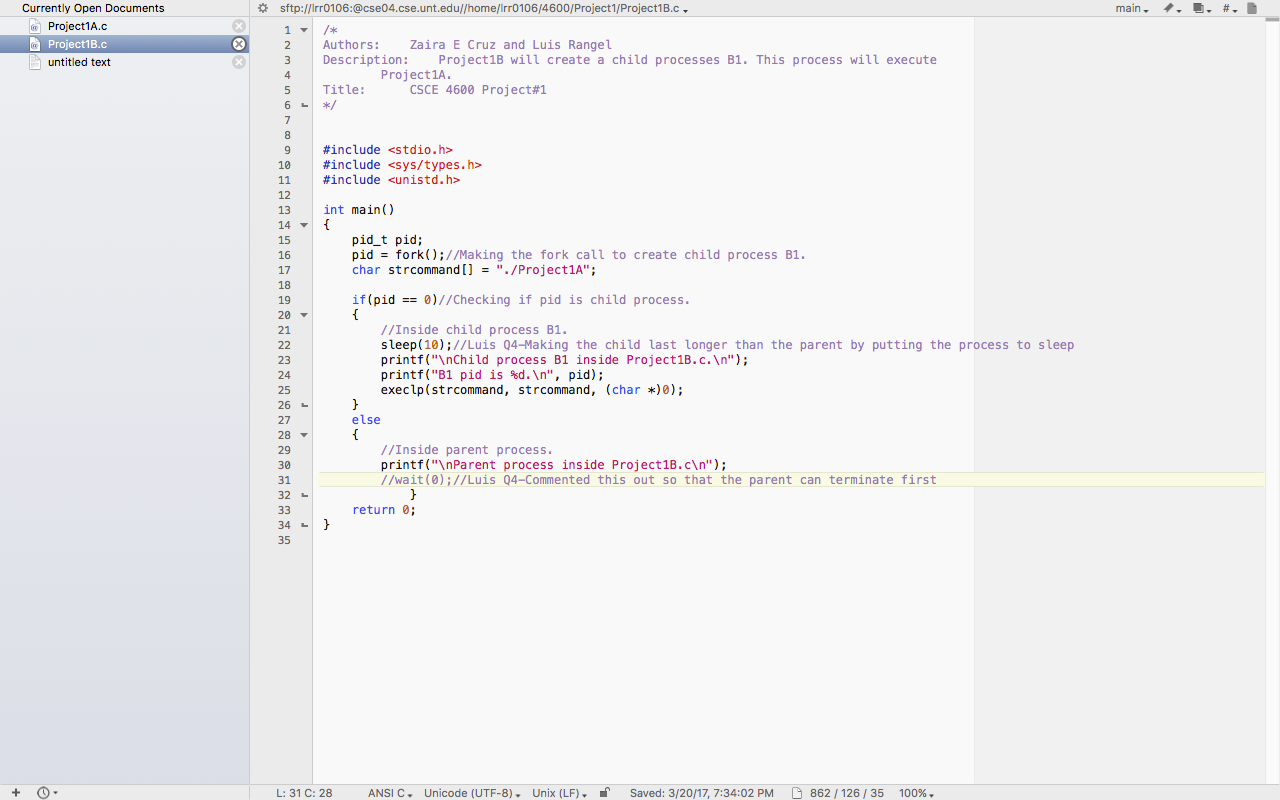
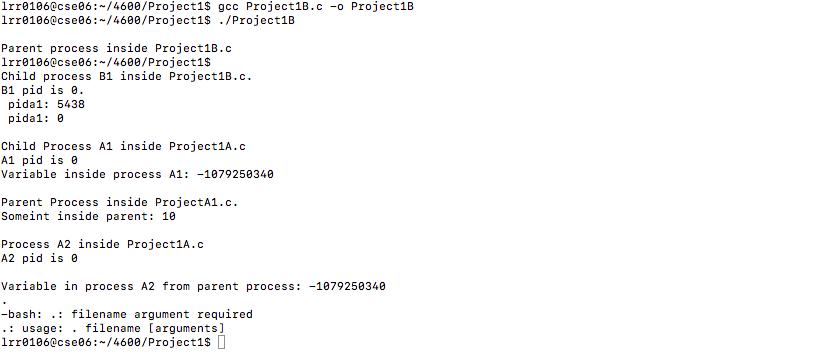
Question 3



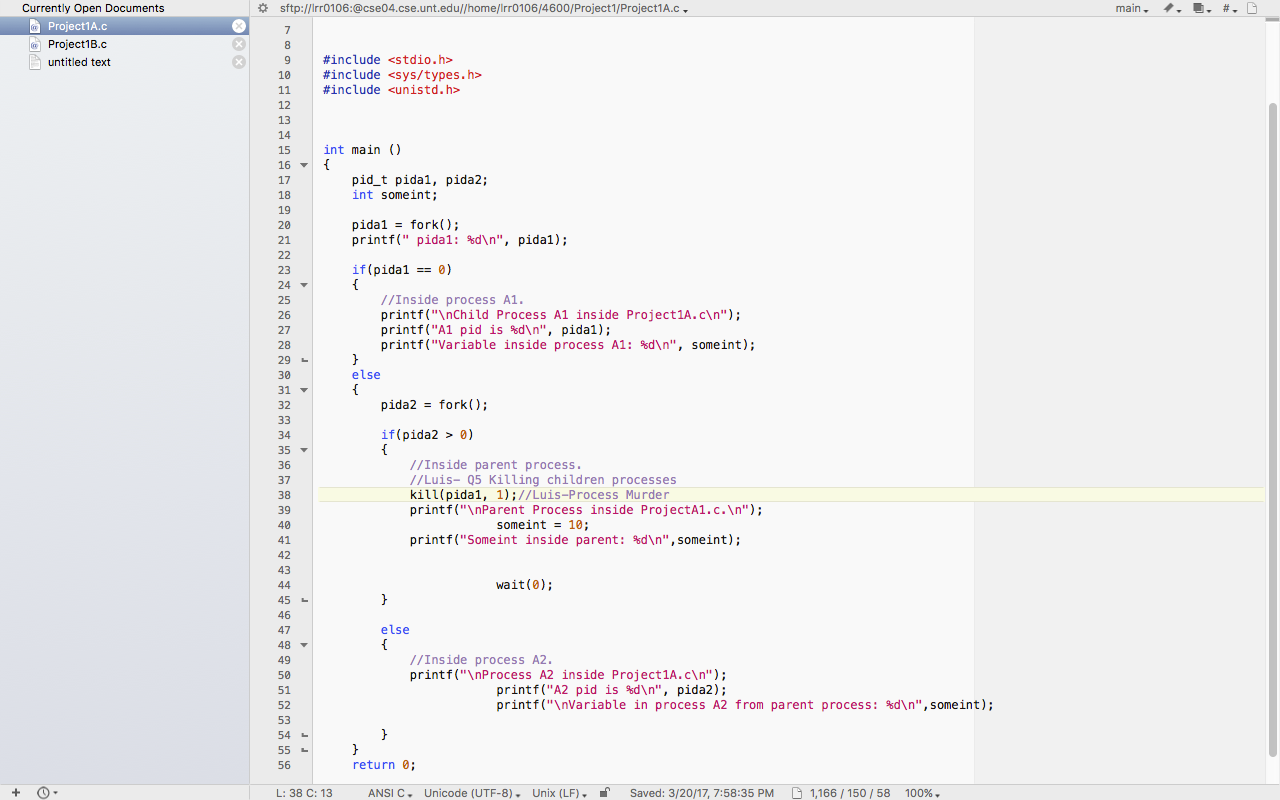
In order to make it so the parent process could not terminate until both it’s children terminate, I added two wait statements at the very beginning of the parent process code block. It was a simple solution and it worked just fine. The console printout shows that the parent executes after the children are done executing!

Question 4



A parent process is allowed to terminate before it’s child are done executing. In order to prove this, I added a sleep(10) call in the child process of Project1B. I also commented out the wait(0) at the end of the parent process, ensuring is no conceivable way the parent process will take longer than the child process. I tested my hypothesis by running Project1B, and the parent process does terminate before the child, shown by being asked for a command after the “Parent process” print statement. After waiting 10 seconds, the print statements for 1B’s child and Project1A are printed out to the screen as normal, showing that the parent’s termination had no effect.

Question 5



Parent processes can kill their children by using the kill() system call. I experimented with this by having the parent in 1A kill child A1 using the kill system call. The console printout shows the results before and after I added the kill system command. As we can see, the “A1 pid is 0” and “Variable inside process A1” all printout just fine before I added the kill command. However, the printouts are missing in the second test, thus showing that the child process was killed off. In order to kill a grandchild, you can use the UNIX kill command to kill off a specific process. You get the grandchild’s process ID and than pass it to the kill system command, then it is dead.