COMP-8567 ASP

Summer 2023

In-Class Lab 3 (Process Control)

a. Write a C program lab3.c to perform the following

   i. Declare three integers num1=10, num2=20, num3=30
   ii. Create a **new file** sample.txt in the **RDWR** mode and assign its file descriptor to fd
   iii. Invoke **three** consecutive fork() system calls
   iv. Each **process** (which is not a direct descendent of the main process) must write "COMP 8567" into sample.txt and then enter a loop that print its PID and PPID continuously with a gap of one sec (15 times)
   v. After exiting the for loop, each **child process** must increment num1, num2 and num3 **(by 10)** , print all the three values and exit the program with exit(0).
   vi. The **parent process** must be made to wait for one of its child processes using the wait() system call.
      1. As soon as the parent process resumes execution after wait(), it writes "HELLO! FROM PARENT" into sample.txt and closes sample.txt.
   vii. Thereafter, the **parent process** must increment num1, num2 and num3 **(by 25)** and print all the three values.

   NOTE: For part b, you are not required to save the output to any file. Just make sure when the GAs run the program to mark it, they can see the intended output.

b. You are required to demonstrate the output in the following scenarios and print appropriate messages related to the execution status of the child process using the macros **WIFEXITED(), WIFSIGNALED(), WEXITSTATUS(), WTERMSIG()**

   i. **Normal exit:** When one of the child processes completes its execution normally, the return value of wait() **and** the exit status must be printed
   ii. **Signaled exit:** You must kill the child process (first child) while it is running using $ kill -9 pid (pid of the child process can be obtained using $ ps -u) , and you must output the signal no associated with the termination of the child process using WTERMSIG()

Submission:

- lab3.c
- Sample.txt