CS 3305B: Operating Systems
Department of Computer Science
Western University
Assignment 1
Winter 2017
Due Date: February 10 2017

Purpose

The goals of this assignment are the following:

- Get experience with the *fork(), wait(), execl(), execlp(), pipe(), and dup2()* system functions
- Learn more about how operating systems are structured and how a shell works
- Gain more experience with the C programming language from an OS perspective

Part I: Parent and Child Processes (40 points)

Write a program in C that will perform the following tasks:

1. Your program will create a *parent* process which will create three *child* processes (e.g., child_1, child_2, and child_3)
2. child_1 will create its own child child_1.1
3. *parent* will wait for child_2 to complete before creating child_3
4. Inside child_3, a system call to an external program will be made (let's say this external program is "B.out" that prints "Hello World"), and as a result of this external program call, child_3 will be replaced by B.out (hint: execl())

The expected output from your program should look like the following:

From *parent* process 2255: child_1 is created with PID 2256
From child_1: child_1.1 is created with PID 2257
From *parent* Process 2255: child_2 is created with PID 2258
From *parent* Process 2255: Waiting for child_2 to complete before creating child_3
From *parent* Process 2255: child_3 is created with PID 2259
From child_3: Calling an external program B.out and leaving child_3…
From the external program B: Hello World..

Hints: fork(), wait(), getpid(), getppid(), execl()

Part II: Inter-Processes Communications (40 points)

Write a C program that will accept two integer values from the user as input (for example, X and Y). Your program will create a *parent* and *child* where the *parent* process will read X and the *child* process will read Y. Now, *parent* and *child* processes will exchange X and

Y by communicating each other through a *pipe* (i.e., shared memory). The expected output from your program should look like the following:

1. From *parent* 2255: *child* with PID 2256 is created
2. From *parent* 2255: Reading X from the user
3. From *child*: Reading Y from the user
4. A pipe is crated for communication between *parent* and *child*
5. From *parent* 2255: Writing X into the *pipe*
6. From *child*: Reading X from the *pipe*
7. From *child*: Writing Y into the *pipe*
8. From *parent* 2255: Reading Y from the *pipe*
9. From *parent* 2255: The value of Y is 10
10. From *child*: The value of X is 100

Hints: fork(), wait(), getpid(), getppid(), pipe(), write(), read()

Part III: I/O Redirection (20 points)

Write a program in C that allows shell command "*ls -l | grep xxxx"* to execute. To do this, your program will create a *parent* and a *child* process. *Parent* process will handle *ls -l* command and the *child* process will handle *grep xxxx* command.

Hints: fork(), wait(), pipe(), dup2(), execlp()

Program Sequence:
1. Create a shared memory *(pipe())*
2. Inside *parent*, perform <stdout> redirection *(dup2())*
3. Inside *parent*, execute *ls -l* command *(execlp())*
4. Inside *child*, perform <stdin> redirection *(dup2())*
5. Inside *child*, execute *grep* command *(execlp())*

Tentative Mark Distribution

This section describes a tentative allocation of marks assigned for the desired features.

- **Part I: Parent and Child Processes (40 points)**
    a) A Parent process will create three Child processes: 10 points
    b) Child_1 will create its own child Child_1.1: 10 points
    c) Parent will wait for Child_2 to complete before creating Child_3: 10 points
    d) Child_3 will make a system call to an external program B.out: 10 points

- **Part II: Inter-Processes Communications (40 points)**
    a) Parent reads X from user: 5 points
    b) Child reads Y from user: 5 points
    c) A pipe is crated for communication between Parent and Child: 5 points
    d) Parent writes X into the pipe: 5 points
    e) Child reads X from the pipe: 5 points
    f) Child writes Y into the pipe: 5 points
    g) Parent reads Y from the pipe: 5 points
    h) Parent and Child prints out the value of X and Y: 5 points

- **Part III: (20 points)**
    a) Create shared memory space (pipe): 4 points
    b) Inside parent, perform <stdout> redirection: 4 points
    c) Inside parent, execute *ls -l* command: 4 points
    d) Inside child, perform <stdin> redirection: 4 points
    e) Inside child, execute *grep* command: 4 points

Assignment related technical resources

Please visit the course website for specific technical instructions and relevant materials. Also, consult TAs, and Instructor for any question you may have regarding this assignment.