Lab 2

CPE 435 – Operating Systems Lab

Noah Woodlee

Start dat: Jan 11

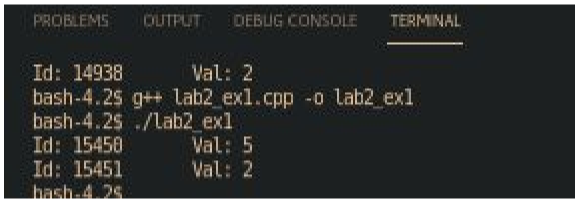
Due date: Jan 25

Theory

1. Process
   A process is a program. This program can do certain tasks such as printing to the terminal, getting user input, or doing operations on files until completion or terminated by the user.
2. Five states of processes
   a. A new process is one that is being created.
   b. A running process is one that is executing instructions
   c. A waiting process is one that is waiting for some event.
   d. A ready process is one that is waiting to be assigned to a process.
   e. A terminated process has finished executing.
3. Orphan process
   An orphan process is still running after its parent has terminated.
4. Zombie process
   A zombie process is a child process that has been terminated but the parent is not aware.
5. Fork()
   Fork creates a child process which is a copy of the parent's process. All statements after fork get executed twice. The value returned for the child is zero and the value returned in the parent is the PID of the new child.
6. Exit()
   Exit can be used to exit a process. Exit(0) exits, while Exit(not 0) fails to exit.
7. Wait()
   Wait can be used to wait on a child to exit. Wait returns the exit status of the child and the PID of the child whose completion caused a wake up.

Procedure and results

Program 1: For program 1 I am using



```
Id: 14938      Val: 2
bash-4.2$ g++ lab2_ex1.cpp -o lab2_ex1
bash-4.2$ ./lab2_ex1
Id: 15450      Val: 5
Id: 15451      Val: 2
bash-4.2$
```

```
using namespace std;
#include <iostream>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>

int main(int argc, char * argv[])
{
int val = 0;
int pid;
pid = fork();
if (pid == 0 ) /* we are the child */
{
val=+2;
cout << "Id: " << getpid() << "\tVal: " << val << endl;
//exit(0);
}
else //(getpid > 0) // we are the parent
{
val=+5;
cout << "Id: " << getpid() << "\tVal: " << val << endl;
//exit(0);
}
}
```