**Lab 3**

**Course: Operating Systems**

*6.1 Questions*

1. What the ouput will be at LINE A?

A: The output will be “Parent: value = 5” because after calling fork(), a new child process is created and loaded into a separate memory location by the operating system. They only share the shared memory segments. As the given code does nothing with shared memory segments, the value in the main process and that in the child process are separate. Thus, when print the “value” value in the main process, it would print it original value (5 in this case).

2. How many processes are created by the program shown below, including the initial parent process? How many process are created when n fork() called?

A: We may know that fork() is a system call that create a child process and both the parents process and the child process is execute after the next line. Thus, after the first fork() calling, there will be 1 child process created (C1). After the second time calling, the initial process (P) will create 1 more child process (C2); the (C1) will create a child (C11). The third time calling, (P) will create one more child (C3); (C1) will create 1 more child (C12), (C11) will create a child (C111); (C2) will create a child process (C21). Thus, there is a total of 8 processes created.

It is easy to see that, after each time calling fork(), the number of the processes created will be doubled. Hence, when n fork() called, there will be 2n processes created.

3. When a process creates a new process using the fork() operation, which of the following states is shared between the parent process and the child process? Why?

A. Stack

B. Heap

C. Shared memory segments

A: A and B are not shared between parents process and child process, while C is. Because: when a process uses fork() to create a child process, a new child process is created and loaded into a separate memory location by the operating system. They shared nothing and have no way to communicate with each other. Thus, an extra piece of memory is attached to the address space for the user to use. As a result, these process share a same space memory segment and have access to it. This is not the case for stack and heap, however, as after creating the child process, new copies of stack and heap are made, when a process tries to write into these stated.

4. What process id (PID) and process group id are used for?

A: Process ID (PID) is used for identifying each running processes in an operating system like Unix, Windows or MacOS. The process group ID is used to identifying each of the process group, a collection of one or more processes. When a child process is created, it belong to its parents process group. The process group is used to control the distribution of the signal, and its ID usually equals the PID of the first member of the group.