**CECS 326 Sec01**

Operating Systems

Elias Woldie (ID 027805724)

Assignment 3

Due Date: 10/19/2023

Submission Date: 10/21/2023

**Program Description**

**1. What all the programs together are designed to do?**

The programs, **master.c** and **slave.c**, collectively aim to demonstrate inter-process communication (IPC) using shared memory in a Linux environment. The main idea is for the master process to spawn a specified number of child processes, which are instances of the slave program. These child processes (slaves) will then communicate with the master process by writing to a shared memory segment. After all child processes complete their execution and communicate their details via shared memory, the master process reads the content of the shared memory, displaying the result of the child processes' activities.

**2. What each individual program does?**

1. **master.c:**

* **Initialization**: Starts by identifying itself with a print statement.
* **Shared Memory Creation**: It creates a shared memory segment with a name obtained from the command-line parameters and structures it as defined in the **myShm.h** header file. This shared memory is initialized with an index set to zero.
* **Spawning Child Processes**: The master process then spawns a specified number of child processes. Each child process will run the slave executable. The master passes the child's number and the shared memory segment name to each child process.
* **Waiting for Child Processes**: After creating the child processes, the master waits until all the children complete their tasks.
* **Display & Cleanup**: Once all child processes terminate, the master reads and displays the content of the shared memory segment, which now contains the child numbers, indicating which child processes have accessed the shared memory. Finally, it removes the shared memory segment and exits.

**slave.c:**

* **Initialization**: On execution, the slave program identifies itself and displays its assigned child number and the shared memory name received from the master process.
* **Access Shared Memory**: The slave opens the existing shared memory segment that was created by the master.
* **Write to Shared Memory**: The slave writes its child number to the next available slot in the shared memory segment. This slot position is determined by the current index value in the shared memory.
* **Closure & Exit**: After writing its child number, the slave process detaches from the shared memory segment and terminates its execution.