Assignment 5

/* You need to put proper explanatory comment in your program to demonstrate the purpose and why you have used the C statements and system calls */

Assignment 5(a):-

Write a C program called Program1 which is responsible for creating a named PIPE or FIFO with necessary permission for all users by using the mknod( ) system call. Now, you keep the facility to read the information from the FIFO if any other process writes some information into the FIFO and as well as keep the facility to write some information into the FIFO, so that Program2 can read it.

Again write another C program called Program2 which is responsible for taking the user input and write it back to the FIFO so that Program1 can read it. Moreover, you have to provide the facility to read the FIFO in this program.

(N.B. - First run the Program1 and keep it in running state in the background. Then run the Program2 which will take user input string and write the input string into the FIFO. Then Program1 will read it from the FIFO and display it and after that Program1 will write something into the FIFO as ACKNOWLEDGEMENT and program2 will read the ACKNOWLEDGEMENT string from the FIFO. In Program2, you may take the input string as a command line argument)

Hints:
* To keep synchronize the Program1 execution with Program2 execution and vice versa, You may need to use the sleep () system call.

Assignment 5(b):-

Write two C programs named program1.c and program2.c to demonstrate the concept of shared memory where program1 (process1) will be responsible for writing its process id and program2 (process2) will be responsible for reading the contents whatever program1 (process1) writes. Moreover, try to remove the shared memory segment that is created for your above mentioned operations properly by using proper system call. Try to demonstrate properly

Hints:

- For creating a shared memory segment or accessing an existing shared memory segment
  you need a system call- **shmget(key_t key, size_t size, int oflag)** .
- For detaches the segment you need a system call- **shmdt(const void *shmaddr)**
- To know how to provide a variety of operations on a shared memory segment, you may go through the system call- **shmctl(int shmid, int cmd, struct shmid_ds *buff)**