Assignment no 6

Problem (A):

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 it's 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 to use system call-**shmget(key_t key, size_t size, int oflag)** .
- For detaching the segment you need to use system call-**shmdt(const void *shmaddr)**
- To know how to implement variety of operations on a shared memory segment, you need to go through the system call-**shmctl(int shmid, int cmd, struct shmid_ds *buff)**
- For better understanding about the system call API, go through the man pages and UNIX NETWORK PROGRAMMING by W.RICHARD STEVENS.
- Check with command *ipcs*

Problem (B):

Write a C program to get the process submission time & termination time of a process. For this assignment your C program will create a child process and writes the submission and termination time of the child process into a file. Learn about *times()* system call. Also learn about use of *gettimeofday()* system call.

Hints:

- The UNIX time(2) is basic time related system call, `time_t time(time_t *t);`, returns the current time in seconds. It returns time as the number of seconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC). If it is non-NULL, the return value is also stored in the memory pointed to by t.
- /* Structure describing CPU time used by a process and its children in sys/times.h*/

```
struct tms
    {   clock_t tms_utime;   /* User CPU time.   */
        clock_t  tms_stime;  /* System CPU time.   */
        clock_t tms_cutime;  /* User CPU time of dead children.   */
        clock_t tms_cstime;  /* System CPU time of dead children.   */

    };
```

- Example program:

```
#include <sys/types.h>
#include <sys/times.h>
main() {

struct tms before, after;    times(&before);
/* ... place code to be timed here ... */    times(&after);
printf("User time: %ld seconds\n", after.tms_utime
        before.tms_utime);    printf("System time: %ld
seconds\n", after.tms_stime         before.tms_stime);
exit(0);
}
```