Grupo 2201 -Alfonso Carvajal, Victor García

Práctica 3

Ejercicio 2:

Explica en que falla el planteamiento de este ejercicio:

En este ejercicio los procesos hijos, que son los que escriben en memoria compartida (también leen para incrementar el id del cliente), se crean en paralelo y no tenemos manera de controlar (sin semaforos) que escriban en orden y le manden la señal al padre que es el que se encarga de leer e imprimir de la memoria compartida de manera que el padre no se salte ninguna escritura. Si el primer proceso escribe, el padre lo lee, y antes de imprimir lo que está en memoria compartida, otro proceso escribe en la memoria compartida, puede darse el caso de que haya una sobreescritura de datos indeseada, de manera que el proceso padre, encargado de imprimir los datos, imprima un mismo dato dos veces. Para esto usaremos semáforos más tarde, y dormir a los hijos un tiempo aleatorio (usleep(pidhijo)) ayuda a que esto no ocurra.

También si no acababa la ejecución de manera correcta, o se interrumpía el programa, la zona de memoria compartida no se libera, y tenemos que hacer esto posteriormente de forma manual desde la terminal.

```
victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $ ./ejercicio2 4

PID: 3258 Introduzca un nombre: Paco

Nombre : Paco
Identidicador: 1

PID: 3276 Introduzca un nombre: Marta

Nombre : Marta
Identidicador: 2

PID: 3298 Introduzca un nombre: María

Nombre : María
Identidicador: 3

PID: 3301 Introduzca un nombre: Alejandro

Nombre : Alejandro
Identidicador: 4
victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $ ipcs

------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x00000000 163840     victor     600        67108864   2          dest
0x00000000 262145     victor     600        524288     2          dest
0x00000000 360450     victor     600        4194304    2          dest
0x00000000 393219     victor     777        4196352    2          dest
0x00000000 557060     victor     600        524288     2          dest
0x00000000 688133     victor     600        524288     2          dest
0x00000000 950278     victor     700        3906760    2          dest
0x00000000 786439     victor     600        2097152    2          dest
0x00000000 983048     victor     600        393216     2          dest
0x00000000 1277961    victor     600        4194304    2          dest
0x00000000 1048586    victor     600        2097152    2          dest

------ Semaphore Arrays --------
key        semid      owner      perms      nsems

victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $
```

Ejercicio 5:

Probamos todas las funciones de la librería de semaforos. Incluimos capturas de la prueba que demuestran que los semaforos se liberan de manera correcta:

```
victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $ ./ejercicio5

Semaforo Creado con id: 655523840

Semaforo Inicializado con valores: 1 1
Valores semaforos despues de UP: 2 1
Valores semaforos despues de DOWN: 2 0
Valores semaforos despues de MULTIPLE UP: 3 1
Valores semaforos despues de MULTIPLE DOWN: 2 0
victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $
```

Podemos observar que efectivamente al inicializar 2 semáforos a 1 y realizar las operaciones:

> Up del primero, Down del segundo, Multiple Up y Multiple Down de ambos, esta es la salida esperada.

```
Semaforo Inicializado con valores: 1 1
Valores semaforos despues de UP: 2 1
Valores semaforos despues de DOWN: 2 0
Valores semaforos despues de MULTIPLE UP: 3 1
Valores semaforos despues de MULTIPLE DOWN: 2 0
victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $ ipcs

------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x00000000 163840     victor     600        67108864   2          dest
0x00000000 262145     victor     600        524288     2          dest
0x00000000 360450     victor     600        4194304    2          dest
0x00000000 393219     victor     777        4196352    2          dest
0x00000000 557060     victor     600        524288     2          dest
0x00000000 688133     victor     600        524288     2          dest
0x00000000 950278     victor     700        3906760    2          dest
0x00000000 786439     victor     600        2097152    2          dest
0x00000000 983048     victor     600        393216     2          dest
0x00000000 1277961    victor     600        4194304    2          dest
0x00000000 1048586    victor     600        2097152    2          dest

------ Semaphore Arrays --------
key        semid      owner      perms      nsems

victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $
```

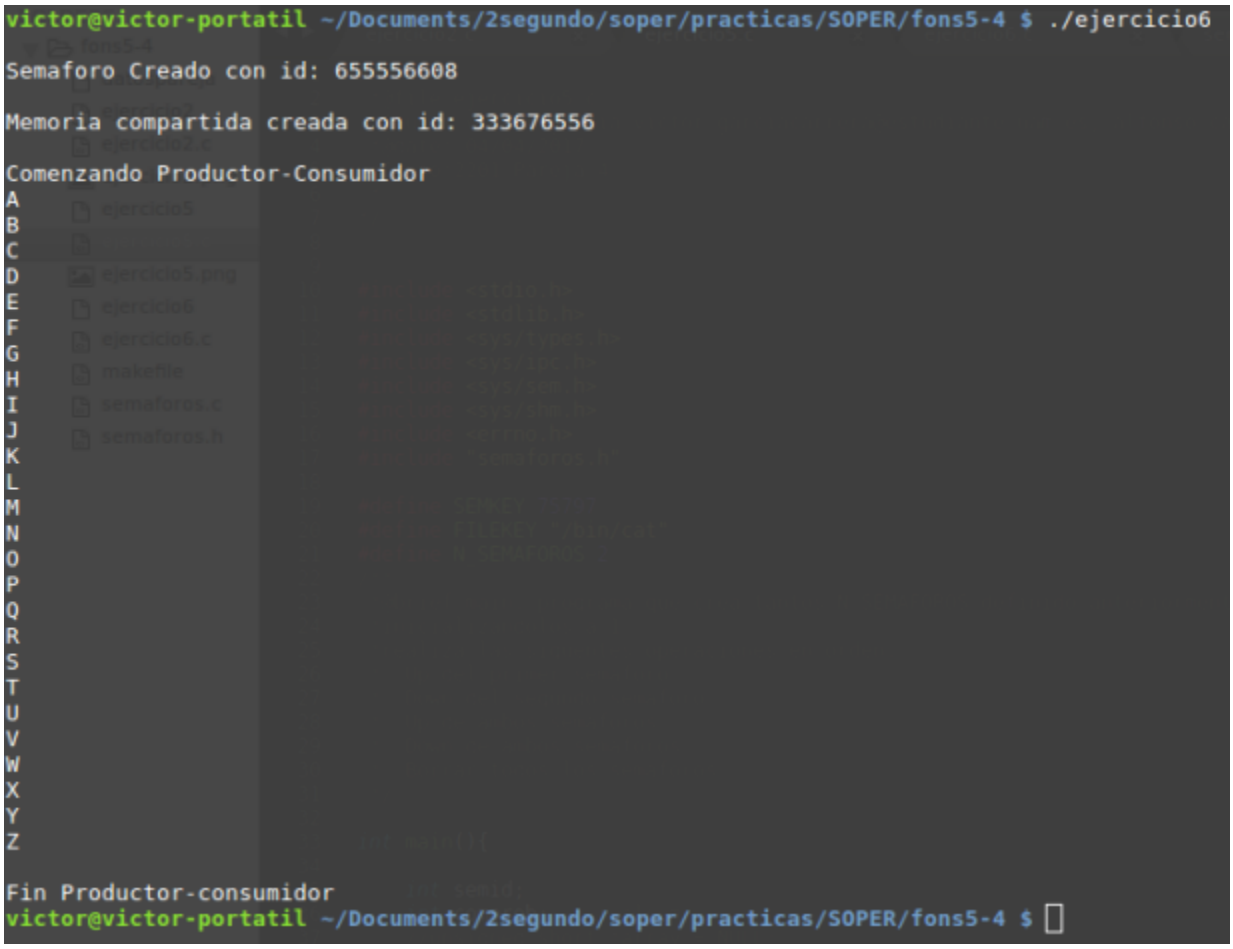
Ejercicio 6:

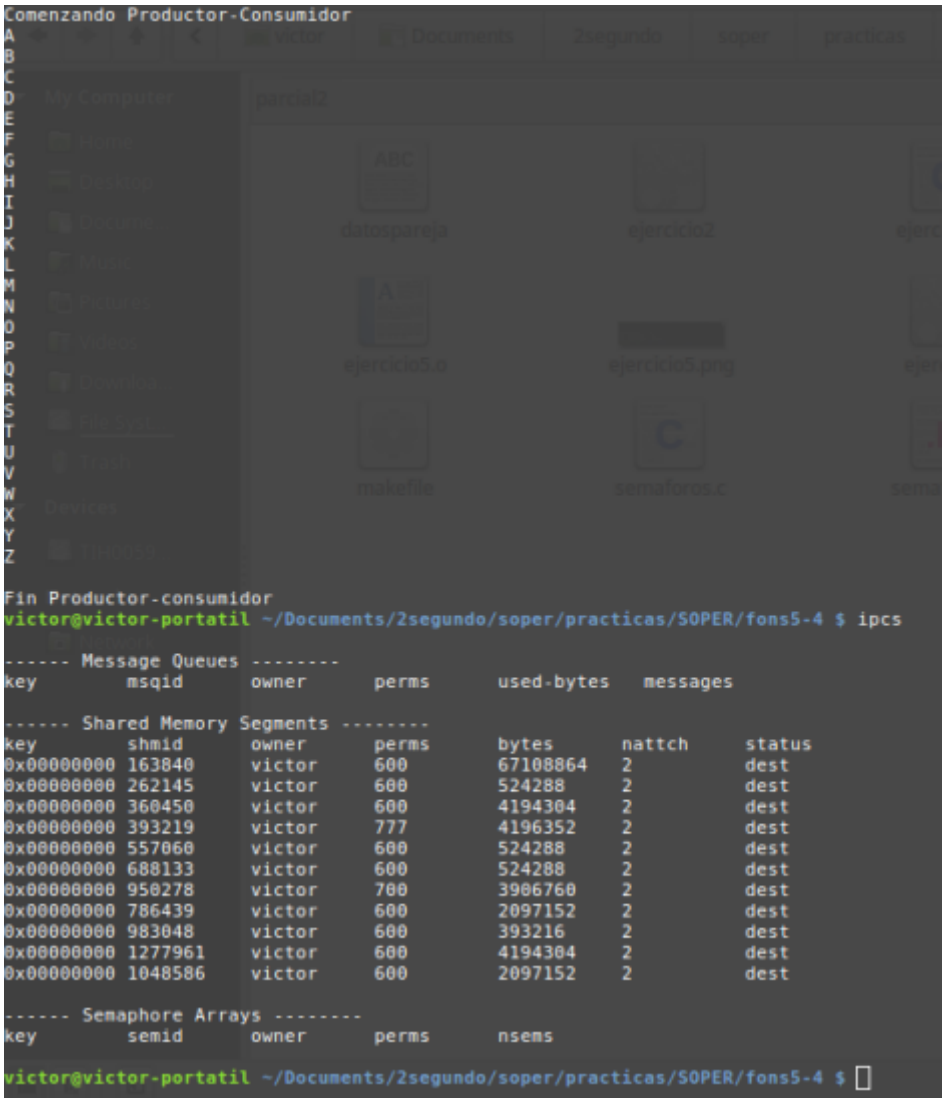
La dificultad fue al acabar el productor, cómo avisar al consumidor de que había finalizado. Esto lo hicimos mediante una condición de salida del bucle while(1) de manera que cuando el productor finalizara, se acabaría, y el padre al leer la ultima letra (“Z”) esperaría a la finalización del hijo.

Utilizamos 3 semáforos para garantizar la concurrencia adecuada de los dos procesos. Uno para proteger la escritura en memoria compartida (productor) y la lectura de memoria compartida (consumidor). Utilizamos otro dos semaforos: en primer lugar para asegurar que el consumidor no intente leer de la memoria compartida sin que haya nada escrita en esta (de manera que se quedarían ambos procesos bloqueados por

hacer un Down del semaforo que protege la sección crítica). En segundo lugar para asegurar que el productor no produzca/escriba otra letra en memoria compartida hasta que el consumidor la haya leido.

Para asegurar que funciona, hemos ejecutado el programa 10.000 veces y en todas las ejecuciones la salida ha sido la esperada.




Comenzando Productor-Consumidor
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
Fin Productor-consumidor
victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $ ipcs
------ Message Queues --------
key msqid owner perms used-bytes messages
------ Shared Memory Segments --------
key shmid owner perms bytes nattch status
0x00000000 163840 victor 600 67108864 2 dest
0x00000000 262145 victor 600 524288 2 dest
0x00000000 360450 victor 600 4194304 2 dest
0x00000000 393219 victor 777 4196352 2 dest
0x00000000 557060 victor 600 524288 2 dest
0x00000000 688133 victor 600 524288 2 dest
0x00000000 950278 victor 700 3906760 2 dest
0x00000000 786439 victor 600 2097152 2 dest
0x00000000 983048 victor 600 393216 2 dest
0x00000000 1277961 victor 600 4194304 2 dest
0x00000000 1048586 victor 600 2097152 2 dest
------ Semaphore Arrays --------
key semid owner perms nsems
victor@victor-portatil ~/Documents/2segundo/soper/practicas/SOPER/fons5-4 $