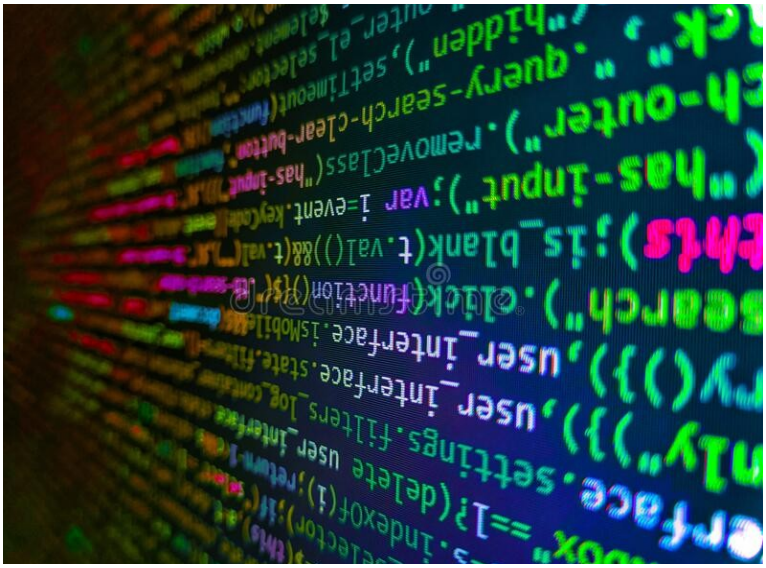
Sistemas Operativos

Práctica 3 – Comandos Para Manipular Ficheros

Semestre de Primavera 2023



Jose Maria Fernández Paredes
Víctor Sort Rubio

Jose Maria Fernández
Víctor Sort

ÍNDICE

1. INTRODUCCIÓN

En esta tercera práctica se nos proponía implementar el juego de “Piedra, papel o tijera” siguiendo un esquema productor – consumidor (con dos productores, que tienen el rol de jugadores, y un único consumidor, con el rol de árbitro).

Aunque la idea de este código puede parecer muy sencilla, se tenían que seguir unas directrices que complicaban la implementación para practicar diversos conceptos vistos en teoría o en teórico-práctica. Se ha trabajado con ficheros csv, señales, búfers y tuberías sin nombre, entre otros recursos.

Al final se nos pedía como extra modificar el código para poder añadir opciones al juego (lagarto y Spock) y hacer una implementación del código sin usar la espera activa.

2. IMPLEMENTACIÓN Y PRUEBAS REALIZADAS

2.1 Implementación básica

Hemos implementado este código siguiendo las instrucciones dadas. El proceso padre hace de productor y ejecuta un código de árbitro, dónde va lanzando señales SIGUSR1 a sus procesos hijos que hacen de consumidores y ejecutan un código de jugador. Cuando los hijos reciben las señales, escriben en una tubería un entero correspondiente a su elección. Cuando el padre las lee, hace la traducción, calcula quien es el ganador, actualiza los contadores y muestra los datos por pantalla. Al haber ejecutado todas las partidas, muestre el resumen global. Este código inicial se ha implementado usando la escucha activa, con una variable global inicializada a 0 cuyo valor se cambia a 1 cada vez que se recibe una señal, para no usar el comando pause que puede dar bastantes problemas.

Una cosa que queremos subrayar de nuestra implementación es que no pasamos los datos del fichero opciones.csv a los productores, pues sólo necesitan saber el número de opciones que tienen, y no cuales son. Se podría haber hecho que ellos devolvieran el string de su elección y el mismo print por pantalla, pero hemos pensado que es mas modular y óptimo de esta manera. También que las funciones competir() y mostrar_opcion() se podrían haber implementado dentro de productor(), pero las sacamos fuera para la mejor comprensión del código, el qual esta muy comentado.

Para las pruebas, inicialmente hicimos todas las comprobaciones pertinentes que hemos ido viendo en las anteriores prácticas. Se comprobó que el programa saliera si no se introducían el número de argumentos esperados, si los ficheros de opciones y reglas no eran válidos o si el número de partidas era 0, negativo o contenía letras (en caso de ser un número positivo no entero, se juegan el número de partidas truncando el valor introducido, por ejemplo, si se introduce 12,758 se juegan 12 partidas).

Para probar su correcto funcionamiento lo ejecutamos con diversos números de partidas distintos, y se comprobó que todas las salidas eran correctas. En el anexo 4.1 se encuentra la salida mostrada por pantalla habiendo jugado 10 partidas, dónde se puede comprobar que las partidas y los resultados se juegan con lógica.

Matemáticamente las probabilidades de ganar, perder o empatar son todas del 33%. Se puede comprobar ejecutando el código con un alto número de partidas que estas probabilidades se

cumplen en la realidad, por ejemplo ejecutando el código 100.000 veces observamos que se aproximan todos los resultados finales a 33.333:

```
oslab:/media/sf_Carpeta_compartida/P3> ./p3.c opciones.csv reglas.csv 100000
...
EL JUGADOR 1 HA GANADO 33373 TURNOS
EL JUGADOR 2 HA GANADO 33341 TURNOS
HA HABIDO 33286 EMPATES
GANA EL JUGADOR 1!!!
```

2.2 Versión extendida del juego

Para hacer la versión extendida no tuvimos que modificar el código, simplemente añadimos las opciones y las nuevas reglas a los ficheros opciones.csv y reglas.csv. Esto demuestra que logramos una muy buena implementación inicial que no dependía de los datos que se encontraran en los ficheros.

Para probar-lo lo ejecutamos con diversos números de partidas distintos, y se comprobó que todas las salidas eran correctas. En el anexo 4.2 se encuentra la salida mostrada por pantalla habiendo jugado 10 partidas, dónde se puede comprobar que las partidas y los resultados se juegan con lógica.

Si ejecutamos el código con 100.000 partidas, cómo hemos hecho antes, observamos que ahora las probabilidades de empate se reducen al 20%, mientras que las de ganar o perder ahora suben hasta al 40%, tal y cómo seria de esperar matemáticamente. Hemos ejecutado este código usando el comando time para poder compararlo con el siguiente punto.:

```
time oslab:/media/sf_Carpeta_compartida/P3> ./p3.c opciones.csv reglas.csv
100000
...
EL JUGADOR 1 HA GANADO 39789 TURNOS
EL JUGADOR 2 HA GANADO 40071 TURNOS
HA HABIDO 20140 EMPATES
GANA EL JUGADOR 2!!!
real  0m9.251s
user  0m8.867s
sys   0m4.762s
```

2.3 Sincronización sin espera activa

Para eliminar la espera activa, creamos dos máscaras de señales, mascara_vacia (claramente, vacía) y mascara_sigusr1 (claramente, incluyendo la señal SIGUSR1). En el main, usando el comando sigprocmask se bloquea la señal SIGUSR1, que después se desbloquea en el código del consumidor (aunque cómo bien explica la guía, no es del todo necesario). Usando sigsuspend en el código del productor se espera perfectamente, y se soluciona el problema del pause.

Hemos vuelto a ejecutar el código con 100.000 partidas, y podemos observar que el tiempo de ejecución i de sistema cae a la mitad, mientras que el de usuario se reduce un 95%.

```
time  oslab:/media/sf_Carpeta_compartida/P3>  ./p3s.c  opciones.csv  reglas.csv
100000
EL JUGADOR 1 HA GANADO 40185 TURNOS
EL JUGADOR 2 HA GANADO 39949 TURNOS
HA HABIDO 19866 EMPATES
GANA EL JUGADOR 1!!!
real  0m5.600s
user  0m0.554s
sys   0m2.460s
```

3. CONCLUSIONES Y VALORACIONES PERSONALES

Cómo se ha visto en el punto anterior, hemos sido capaces de implementar todo lo requerido y funcionan correctamente en todos los posibles casos, considerando también mensajes de error. Por esta parte pues, estamos muy satisfechos con nuestros resultados.

Además, realmente creemos que la realización de esta práctica nos ha aportado mucho conocimiento referente a todos los conceptos con los cuales hemos trabajado, especialmente los referentes a señales y a sus máscaras, usadas en el ejercicio opcional.

Finalmente, creemos que hemos trabajado muy bien en equipo, que nos hemos ayudado mutuamente y esforzado para aprender el temario que tocaba, a la vez que sinceramente nos lo hemos pasado bien programando el código. En resumen, y en nuestra opinión, hemos hecho una muy buena práctica.

Jose Maria Fernández
Víctor Sort

4. ANEXOS

4.1 Salida por pantalla de la implementación básica

```
oslab:/media/sf_Carpeta_compartida/P3> ./p3.c opciones.csv reglas.csv 10
* TURNO 1
El Jugador 1 ha escogido Papel
El Jugador 2 ha escogido Tijeras
Tijeras cortan Papel
Gana el Jugador 2!
* TURNO 2
El Jugador 1 ha escogido Piedra
El Jugador 2 ha escogido Papel
Papel envuelve Piedra
Gana el Jugador 2!
* TURNO 3
El Jugador 1 ha escogido Piedra
El Jugador 2 ha escogido Tijeras
Piedra aplasta Tijeras
Gana el Jugador 1!
* TURNO 4
El Jugador 1 ha escogido Tijeras
El Jugador 2 ha escogido Piedra
Piedra aplasta Tijeras
Gana el Jugador 2!
* TURNO 5
El Jugador 1 ha escogido Piedra
El Jugador 2 ha escogido Piedra
Empate!
* TURNO 6
El Jugador 1 ha escogido Papel
El Jugador 2 ha escogido Piedra
Papel envuelve Piedra
Gana el Jugador 1!
* TURNO 7
El Jugador 1 ha escogido Tijeras
El Jugador 2 ha escogido Tijeras
Empate!
* TURNO 8
El Jugador 1 ha escogido Piedra
El Jugador 2 ha escogido Piedra
Empate!
* TURNO 9
El Jugador 1 ha escogido Papel
El Jugador 2 ha escogido Papel
Empate!
* TURNO 10
El Jugador 1 ha escogido Papel
El Jugador 2 ha escogido Papel
Empate!
EL JUGADOR 1 HA GANADO 2 TURNOS
EL JUGADOR 2 HA GANADO 3 TURNOS
```

Jose Maria Fernández
Víctor Sort

```
HA HABIDO 5 EMPATES
GANA EL JUGADOR 2!!!
```

4.2 Salida por pantalla de la versión extendida del juego

```
oslab:/media/sf_Carpeta_compartida/P3> ./p3.c opciones.csv reglas.csv 10
* TURNO 1
El Jugador 1 ha escogido Piedra
El Jugador 2 ha escogido Lagarto
Piedra aplasta Lagarto
Gana el Jugador 1!
* TURNO 2
El Jugador 1 ha escogido Papel
El Jugador 2 ha escogido Piedra
Papel envuelve Piedra
Gana el Jugador 1!
* TURNO 3
El Jugador 1 ha escogido Tijeras
El Jugador 2 ha escogido Spock
Spock rompe Tijeras
Gana el Jugador 2!
* TURNO 4
El Jugador 1 ha escogido Tijeras
El Jugador 2 ha escogido Lagarto
Tijeras decapitan Lagarto
Gana el Jugador 1!
* TURNO 5
El Jugador 1 ha escogido Lagarto
El Jugador 2 ha escogido Piedra
Piedra aplasta Lagarto
Gana el Jugador 2!
* TURNO 6
El Jugador 1 ha escogido Piedra
El Jugador 2 ha escogido Piedra
Empate!
* TURNO 7
El Jugador 1 ha escogido Spock
El Jugador 2 ha escogido Lagarto
Lagarto envenena Spock
Gana el Jugador 2!
* TURNO 8
El Jugador 1 ha escogido Lagarto
El Jugador 2 ha escogido Spock
Lagarto envenena Spock
Gana el Jugador 1!
* TURNO 9
El Jugador 1 ha escogido Tijeras
El Jugador 2 ha escogido Tijeras
Empate!
* TURNO 10
El Jugador 1 ha escogido Spock
```

Jose Maria Fernández
Víctor Sort

```
El Jugador 2 ha escogido Lagarto
Lagarto envenena Spock
Gana el Jugador 2!
EL JUGADOR 1 HA GANADO 4 TURNOS
EL JUGADOR 2 HA GANADO 4 TURNOS
HA HABIDO 2 EMPATES
EMPATE!!!
```