SEGUNDA PRÁCTICA SSOO I 2021

ENUNCIADO:

El programa que hay que presentar constará de un único fichero fuente de nombre `pistolos.c`. La correcta compilación de dicho programa, producirá un fichero ejecutable, cuyo nombre será obligatoriamente `pistolos`. Respetad las mayúsculas/minúsculas de los nombres, si las hubiere.

La ejecución del programa creará una serie de procesos que representarán una partida del *problema del pistolero*.

En dicho problema, un número determinado de pistoleros del oeste norteamericano se colocan en círculo. Cada uno de ellos elige al azar a un compañero. A la de tres, disparan todos al unísono. Algunos pistoleros se libran, algunos reciben uno o más disparos. Se retiran los cadáveres, los supervivientes cierran el corro y comienza una nueva ronda de elección y disparos. La historia puede acabar de dos únicos modos, con la supervivencia de uno o de ningún pistolero. Matemáticamente, lo interesante es calcular la probabilidad de que sobreviva un pistolero dependiendo del número inicial de ellos. Esto no nos interesará en esta práctica.

En la práctica, cada pistolero vendrá representado por un proceso, hijos todos del proceso original, que ejecuta `main`. A este proceso original se le pasará por la línea de órdenes un número entero, correspondiente con el número inicial de pistoleros que forman la partida. El número estará comprendido entre 3 y 128.

A continuación, tendrá tantos hijos como indique dicho número. El padre regula el funcionamiento de las rondas hasta llegar al final de ellas. En el caso de que, al final, quede un proceso pistolero vivo, lo matará, tomará su código de retorno y pondrá el resultado final de la partida. El padre sólo matará, de tener que hacerlo, a este último pistolero. No participa en las rondas disparando.

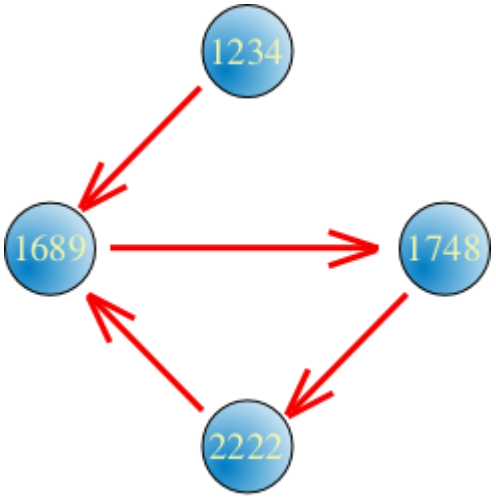
FUNCIONAMIENTO RONDA DE DISPAROS

En cada ronda de disparos, el proceso padre habrá bloqueado previamente un fichero donde guarda los PIDs de los pistoleros que están vivos para que ellos lo puedan leer, una vez desbloqueado el fichero, y sepan a quién pueden disparar. El padre escribe en la pantalla lo equivalente a este ejemplo:

```
*** COMIENZA LA RONDA NUMERO 7 ***
*** PIDs: 1234 1689 1748 2222
```

Acto seguido, desbloquea el fichero para que cada uno de los pistoleros puedan elegir a un compañero y dispararle.

Para ello, los pistoleros eligen a otro pistolero al azar. No pueden dispararse a sí mismos. Anuncian el PID del pistolero elegido por la pantalla y envian una señal `SIGTERM` a dicho pistolero, para matarlo.



La salida por la pantalla siguiendo el ejemplo anterior podría ser:

```
1234->1689
2222->1689
1748->2222
1689->1748
```

Debe aparecer, por lo tanto, una línea por cada proceso indicando a qué proceso ha disparado. No tienen por qué aparecer en orden. Dependerá de cómo reparta el Sistema Operativo la CPU. Y también deben disparar todos los pistoleros, incluso aunque les haya llegado un disparo antes.

El proceso padre hará tantas llamadas al sistema `wait`s como procesos pistoleros había antes de disparar y, de este modo, sabrá qué procesos han muerto a causa de los disparos.

Como puede que algún proceso quede vivo y, para que el proceso padre no espere eternamente, se necesita algo que le despierte de la espera bloqueante en que le sumen los `wait`s. Esto lo logra el propio proceso padre, mediante una llamada previa a a `alarm` antes de dormirse. La llamada hará que el Sistema Operativo le envíe una señal `SIGALRM` cuando pase un segundo.

El proceso padre sabe, al final, cuántos procesos han quedado vivos y puede iniciar una nueva ronda en el caso de que sea necesario.

En el caso de que haya que hacer una nueva ronda, para indicar a los supervivientes que pueden continuar, el padre les enviará una señal `SIGUSR1`. Los hijos responderán al padre también con una señal `SIGUSR1`. Mientras queden hijos por responder, el padre no continúa con la siguiente ronda.

RESTRICCIONES

- Se deberán usar llamadas al sistema siempre que sea posible, a no ser que se especifique lo contrario.

- No está permitido usar la función de biblioteca `system`, salvo indicación explícita en el enunciado de la práctica.

- No se puede suponer que los PIDs de los procesos de una ristra van a aparecer consecutivos. Puestos en plan exquisito, ni siquiera podemos suponer que estarán ordenados de menor a mayor (puede ocurrir que se agoten los PIDs y retome la cuenta partiendo de cero).

- No está permitido el uso de ficheros, tuberías u otro mecanismo externo para transmitir información entre los procesos, salvo que se indique en el enunciado.

- Supondremos un límite máximo de pistoleros iniciales igual a 128 procesos. Este límite os puede servir para no tener que usar memoria dinámica si no lo deseáis.