Examen P1 y P2 - 2020

1. Ejercicio Práctica 1: (Semáforos). Problema de los fumadores

- Los fumadores, inmediatamente después de fumar, llaman a una función nueva que se llama “tirar_colilla” (simula la acción de tirar la colilla a una papelera).

- Habrá una serie de nuevas hebras, llamadas hebras recolectoras. Define una constante con el número de dichas hebras, que debe ser divisor del número de fumadores. Cada hebra recolectora tiene asociada una papelera distinta. Las hebras recolectoras (y por tanto las papeleras) se numeran empezando en 0. Cada papelera tiene asociado un número de colillas en la misma (inicialmente cero).

- Las hebras recolectoras ejecutan un bucle infinito, en cada iteración se bloquean esperando que su papelera esté llena, entonces son despertadas por un fumador, imprimen un mensaje, vacían la papelera (ponen su número de colillas a cero), y avisan al fumador que las ha despertado de que ya la han vaciado (ver siguiente punto).

- El fumador número i usa la papelera cuyo número es el resto de la división entera de i entre el número de papeleras. En la función “tirar_colilla” , cada fumador debe tirar una colilla a su papelera, lo que significa: (a) incrementar el número de colillas de su papelera y después (b), si ese número ya ha llegado a 4, (la papelera está llena), debe despertar a la hebra recolectora asociada a esa papelera y esperar a que la recolectora la vacíe.

Ten en cuenta que mientras un fumador está tirando una colilla a su papelera o esperando a que su papelera sea vaciada, ningún otro fumador puede tirar ninguna colilla a esa misma papelera (pero sí a otras).

2. Ejercicio Práctica 2: (Monitores). Problema de los lectores/escritores usando monitor SU

- En este programa queremos comparar el orden de llegada con el orden de acceso al recurso. El orden de llegada es el orden en el que las hebras logran iniciar los métodos ini..., y el orden de acceso al recurso es el orden en el que las hebras terminan de ejecutar esos métodos (independientemente del rol). Para lograr esto, se usa una variable contador (lleva el orden de llegada), inicializada a 0, que se va incrementando en cada llegada (al inicio de ini...), y después se copia sobre otra variable local al método. La variable local se imprime al final del método (cuando la hebra ya tiene garantizado el acceso para leer o escribir en el recurso compartido).

- En esta nueva versión queremos intentar que las hebras accedan al recurso en el orden de llegada, es decir, el orden en el que inician la ejecución de los métodos ini..., independientemente de su rol. Se usan dos variables condición (en lugar de las originales), llamadas resto y puerta. En la puerta únicamente esperará como mucho una hebra hasta que le sea posible acceder al recurso. El resto de hebras que estén esperando, esperarán en la cola resto hasta que la puerta quede libre (la cola resto es previa a la puerta).

- Al inicio de los métodos ini... (después de incrementar el contador y conocer su orden de llegada), cada hebra comprueba si hay alguna otra hebra esperando en la puerta, en ese caso espera en la cola resto. Después espera bloqueada en la puerta mientras que no puedan acceder al recurso. Aunque la hebra en la puerta se desbloquee, debe volver a comprobar si puede acceder (en bucle), y si no puede debe volver a bloquearse en la puerta. Cuando finalmente pueden acceder (al para que acceda a la puerta.

- Al final de los métodos fin... (cuando ya han terminado de usar el recurso), las hebras desbloquean a la hebra en la puerta (si hay alguna), para que dicha hebra en la puerta pueda comprobar si ahora ya puede efectivamente acceder al recurso.