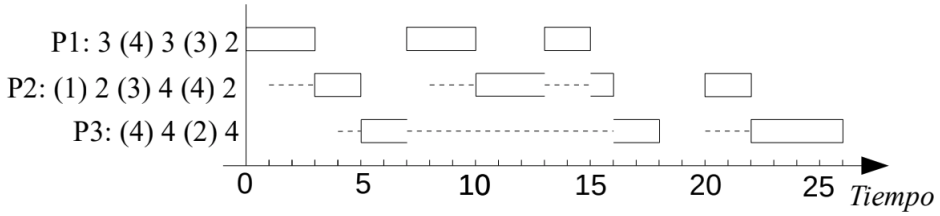
CC4302 – Sistemas Operativos
Auxiliar 6

Profesor: Luis Mateu
Auxiliar: Diego Madariaga

29 de abril de 2020

1. P1.b Control 2 2019/2

El diagrama de abajo muestra las decisiones de scheduling para 3 procesos. A la izquierda se indica para cada proceso las duraciones de sus ráfagas de CPU y entre paréntesis las duraciones de sus estados de espera. En el diagrama la línea punteada indica que el estado del proceso es READY. Si el proceso está en estado de espera el espacio aparece en blanco.



¿De qué estrategia de scheduling se trata? Rehaga el diagrama completo considerando ahora la estrategia *first come first served*.

2. Impresora Server

Suponga que en nSystem se dispone de una impresora compartida por todas las tareas. Dos o más tareas pueden usar simultáneamente la impresora, pero en este caso las líneas de la impresora saldrían entremezcladas. En una aplicación se desea evitar el entremezclado de líneas haciendo que cada tarea solicite el acceso exclusivo a la impresora antes de ocuparla y además notifique cuando termina de utilizarla. Por lo tanto una tarea tendrá la siguiente forma:

```
tarea() {
    ...
    obtenerImpresora();
    ...                    /* utilizar impresora */
    devolverImpresora();
    ...
}
```

Además, por razones de ahorro de electricidad, se necesita que la impresora se coloque en modo de bajo consumo cada vez que transcurran 5 minutos sin ser utilizada por ninguna tarea. Para colocar la impresora en modo de bajo consumo invoque el procedimiento `modoBajoConsumo()`. Para volver a usar la impresora cuando está en modo de bajo consumo, invoque el procedimiento `modoUsoNormal()`.

Implemente los procedimientos `obtenerImpresora()`, `devolverImpresora()` y un procedimiento `inicializarImpresora()` que se invoca al inicio de `nMain`. El uso de busy-waiting o consultas periódicas para ver si se han cumplido los 5 minutos es equivalente a no responder la pregunta.

3. Mensajería en nSystem

1. Implemente los métodos de nSystem del sistema de mensajería sin soportar timeouts.
   - `int nSend(nTask task, void *msg)`
   - `void *nReceive(nTask *ptask, int timeout)`
   - `void nReply(nTask task, int rc)`
2. Modifique la implementación anterior para que SÍ soporte timeouts.