EJERCICIOS TEMA 5 PROGRAMACION CONCURRENTE

FECHA REALIZACIÓN:
NOMBRE:
NOTA:

Asume que hay dos operaciones `request(units)` y `release(units)`, donde `units` es un número positivo. Cuando un proceso llama a `request`, se retrasa hasta que haya al menos `units` páginas de memoria disponibles. Un proceso devuelve `units` páginas de memoria al hacer `release`. Las páginas se pueden devolver en cantidades diferentes a las que se solicitaron. Se pide:

1. Desarrolla un monitor que implemente `request` y `release`. Especifica el invariante global. No debes preocuparte del orden en el que los `request` se sirven. Utiliza la disciplina SC.

2. Modifica la solución en el apartado 1 para utilizar una política de asignación de recursos *shortest-job-next* de manera que los requests más pequeños tengan preferencia sobre los más grandes.

3. Modifica la solución en el apartado 1 para utilizar una política de asignación de recursos *first-come first-served*. Esto implica que un request pendiente puede tener que retrasarse incluso aunque haya suficiente memoria disponible.

4. Asume que request y release adquieren y devuelven páginas adyacentes de memoria; es decir, si un proceso pide dos páginas, se retrasa hasta que pueda conseguir dos páginas adyacentes. Desarrolla un monitor que implemente esta versión de request y release. Asume que el estado de la memoria viene representado por un array de enteros mem de tamaño n tal que $mem[i]$ vale 0 si la posición de memoria está libre, y vale 1 si está ocupada.