
 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos</p> <p>Examen de la convocatoria extraordinaria 21 de junio de 2011</p>	
---	--	---

ATENCIÓN:

- Lea atentamente todo el enunciado antes de comenzar a contestar.
- Dispone de 3 horas para realizar la prueba.
- No se podrán utilizar libros ni apuntes, ni calculadoras de ningún tipo.
- Los teléfonos móviles deberán permanecer desconectados durante la prueba (apagados, no silenciados).
- Solamente se corregirán los ejercicios contestados con bolígrafo. Por favor no utilice lápiz.

Ejercicio 1. Teoría [2,5 puntos]:

Pregunta 1. Haga un esquema y explique la estructura de un sistema operativo por capas, como Windows.

Pregunta 2. Explique cómo se trata en Linux una señal como SIGINT.

Pregunta 3. ¿Cuál es la diferencia entre un semáforo de Linux y un mutex?

Ejercicio 2 [2,5 puntos]:

Se desea realizar una aplicación en C que conste de un proceso padre y varios hijos.

El proceso padre debe, cada vez que reciba la señal SIGUSR1 crear un proceso hijo y mostrar cuantos hijos están vivos en ese momento. El padre estará en un bucle infinito esperando que finalicen los hijos o que se reciba la señal.

Los hijos deben estar vivos durante un tiempo aleatorio entre 5 y 20 segundos y después deben finalizar.

Se pide:

1. Realizar el código del proceso hijo.
2. Programar el código del proceso padre que incluirá:
 - El tratamiento de la señal de CTRL-C y la creación de los hijos
 - El programa principal con un bucle que espere por los hijos que han terminado.



Ejercicio 3 [2,5 puntos]:

El siguiente programa ofrece una solución basada en semáforos para el problema de productor-consumidor

```
int BufferSize = . . . ;

semaphore mutex = . . . ;
semaphore empty = . . . ;
semaphore full = . . . ;

producer()
{
    int item;
```

 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos</p> <p>Examen de la convocatoria extraordinaria 21 de junio de 2011</p>	
---	--	---

```

while (TRUE) {
    make_new(item);
    down(&empty);
    down(&mutex);
    put_item(item);
    up(&mutex);
    up(&full);
}

}

consumer()
{
    int item;

    while (TRUE) {
        down(&full);
        down(&mutex);
        remove_item(item);
        up(&mutex);
        up(&empty);
        consume_item(item);
    }
}

```

Se pide:

- Inicialice la variable BufferSize y los 3 semáforos, asumiendo que el buffer tiene una capacidad máxima de almacenamiento de 10 objetos.
- ¿Es el semáforo mutex necesario? Justifique su respuesta.
- ¿Es posible simplificar el programa utilizando solamente un semáforo count en vez de los semáforos full and empty? Justifique su respuesta. Si es posible, describa la solución.
- En la solución proporcionada, hay algún problema si se intercambian las llamadas down(&empty) y down(&mutex) en el productor?

Ejercicio 4 [2,5 puntos]:

Se tiene un disco con sistema de ficheros Unix. El tamaño del bloque de ficheros es de 1KByte, las direcciones a los bloques son de 4 bytes y los i-nodos tienen la estructura tradicional (10 apuntadores directos, 1 indirecto simple, 1 indirecto doble y 1 indirecto triple). Se pide:

- Escribir un programa que lea 8200 KBytes con acceso a nivel de 1 KByte.
- Calcular cuántos bloques de disco incluyendo el i-nodo y los bloques de direcciones y datos se estarán utilizando para almacenar un fichero de 8200 KBytes. Explique detalladamente cómo se han realizado los cálculos.