



Apellidos, Nombre: \_\_\_\_\_

DNI: \_\_\_\_\_

### Cuestiones

C1.- (1.5pts) Indique cuáles de las siguientes afirmaciones son verdaderas y cuáles falsas. En el caso de que considere que la respuesta es falsa, indique el motivo.

- Un driver se compila como un programa normal, la única diferencia es que debería ejecutarlo un usuario con privilegios.
- Un SO tipo UNIX utiliza el major number de un dispositivo para identificar al driver que lo gestiona.
- Un proceso de usuario sólo puede realizar operaciones de Entrada/Salida solicitándolas a través de una interrupción software (`trap`).
- Cuando un módulo del kernel termina, basta con que éste haga `exit()` para descargarse.
- Si un módulo del kernel realiza una operación no permitida, como por ejemplo acceso a una región de memoria no disponible, el kernel lo detecta y descarga el módulo.

C2.- (1.5pts) En un sistema tipo UNIX, un proceso ejecuta el siguiente programa que pretende dividir en dos un fichero utilizando dos procesos concurrentes:

```
1 int main(int argc, char *argv[]) {
2     int fd0, fd1, fd2, mid, ret;
3     struct stat stBuf;
4     int flags=O_WRONLY|O_TRUNC|O_CREAT;
5
6     if( (fd0=open("origen",0)) ==-1 ||
7         (fd1=open("mitad1", flags)==-1 ||
8         (fd2=open("mitad2", flags)==-1 )
9         perror(), exit(-1);
10
11     stat("origen.txt", &stBuf);
12     mid=stBuf.st_size/2;
13
14     if( fork() == 0 ){
15         lseek(fd0, mid, SEEK_SET);
16         ret=copia(fd0,fd2,stBuf.st_size-mid);
17     }
18
19     else{
20         lseek(fd0, 0, SEEK_SET);
21         ret=copia(fd0, fd1, mid);
22     }
23     return(ret);
24 }
25 int copia(int fd0, int fdD, int nB){
26     int i; char dato;
27     for(i=0;i<nB;i++){
28         if(read(fd0,&dato,1) !=1 ||
29            write(fdD,&dato,1) !=1 ){
30             perror("Error");
31             return(-1);
32         }
33     }
34     return(0);
35 }
```

- ¿Los ficheros `mitad1` y `mitad2` se crearán correctamente en todos los casos? ¿Por qué?
- ¿Qué modificaciones mínimas propone para arreglar el código y que la copia  **siga siendo concurrente**?

C3.- (1pt) Considerar la siguiente secuencia de referencias a memoria virtual generadas por programa en un sistema con paginación pura:

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

Cartagena99

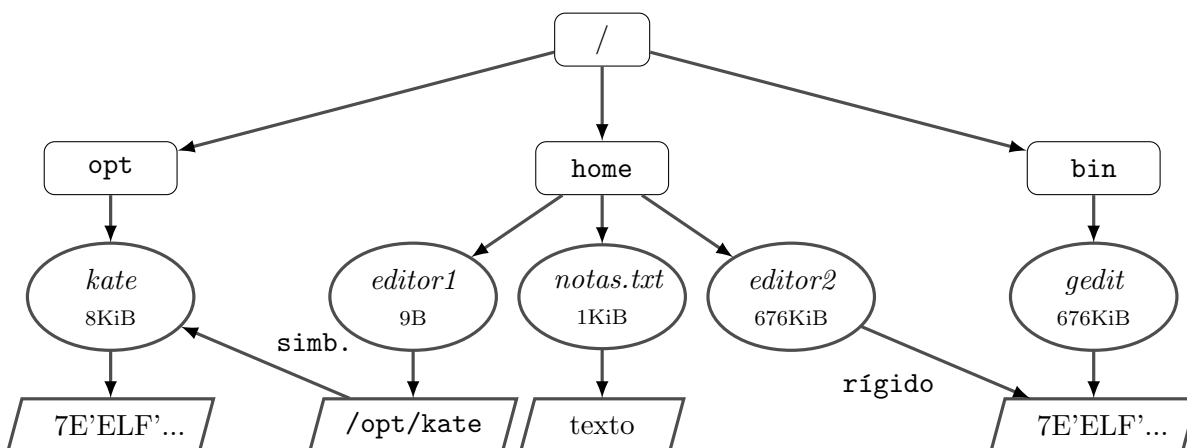
## Problemas

**P1.-** (1.5 pts) En un sistema monoprocesador, un usuario lanza tres trabajos intensivos en CPU (A, B y C) al mismo tiempo. La estimación de los tiempos de CPU de los procesos es de 5, 3 y 7 unidades de tiempo, respectivamente. Para cada uno de los siguientes algoritmos de planificación determine los tiempos de ejecución y de espera de cada proceso y la productividad.

- SJF
- RR con  $q = 2$
- RR con  $q \approx 0$ , es decir, empleando la política de compartición ideal del procesador y midiendo los tiempos cuando finaliza cada proceso.

**P2.-** (2.5 pts) Un sistema de ficheros UNIX utiliza bloques de 2KiB y direcciones de disco de 32 bits. Los nodos-i contienen 10 direcciones de disco para bloques de datos, una dirección de bloque indirecto simple, una dirección de bloque indirecto doble.

- ¿Cual será el tamaño máximo de un fichero en este sistema suponiendo despreciable el espacio ocupado por el superbloque y la tabla de nodos-i?
- Siguiendo el siguiente árbol de directorios:
  - Indique una posible asignación de números de nodos-i.
  - Indique el contenido del nodos-i correspondiente al directorio `home`, así como de todos los nodos-i referenciados por él. Muestre también el contenido de los bloques de datos.



**P3.-** (2 pts) Problema del barbero dormilón con dos barberos. En la barbería hay una sala de espera con  $n$  sillas y, en este caso, dos barberos. El comportamiento de los clientes y barberos será el siguiente:

- Si entra un cliente en la barbería y todas las sillas están ocupadas, el cliente abandona la barbería.
- Si hay sitio, el cliente ocupa una silla y decide qué barbero le atenderá. Llevaremos el registro de cuántos clientes hay esperando a cada barbero mediante `n_clientes[idx]` y elegiremos al que menos tenga (en caso de empate, podemos hacer `rand()%2`).
- Si un barbero no tiene a ningún cliente esperando en su cola se va a dormir (`dormido[idx]=1`).
- Si el barbero estaba dormido, el primer cliente en su cola de espera le despertará.
- Una vez que el cliente va a ser atendido, el barbero invocará `cortar()` y el cliente `sentarse()`

Escriba un programa que coordine al barbero y a los clientes utilizando mutex y semáforos POSIX

**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

---

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70**

Cartagena99

```

//despertando al barbero si
//fuese necesario
sentarse();

```