En un monoprocesador con sistema operativo de propósito general, durante un período de tiempo, se están ejecutando de forma concurrente los siguientes procesos, además del proceso nulo:

- Proceso 1: Servicio de *spooling*. Se trata de un demonio, que está esperando las peticiones de impresión de los usuarios. Este proceso tiene prioridad 5.
- Proceso 2: Proceso perteneciente al usuario *Pepe* que ejecuta el código descrito en el cuadro 1. El descriptor `fd` corresponde a un fichero que se ha abierto anteriormente de forma correcta y que contiene 10 bytes de información. Este proceso tiene prioridad 0.
- Proceso 3: Proceso perteneciente al superusuario que ejecuta el código descrito en el cuadro 2. La variable `pid` contiene el identificador del proceso 3. El proceso 3 tiene prioridad 3.
- Proceso 4: Proceso perteneciente al usuario *Juan* que ejecuta el mandato `lpr /home/juan/datos.txt`. Este proceso tiene prioridad 0.

Cuadro 1: Código del Proceso 2:

```
1. switch(fork()){
2. case 0: read(fd, buffer, 10);
3.         exit(0);
4. default: wait();
5.         fprintf("%d\n", 4/0);
6.         exit(0);
7. }
```

Cuadro 2: Código del Proceso 3:

```
1. void manejador( int sig ) {
2.     kill(pid, SIGKILL);
3. }

4. int main( int argc, char *argv[] )  {
5.     struct sigaction sact;

6.     sigemptyset( &sact.sa_mask );
7.     sact.sa_flags = 0;
8.     sact.sa_handler = manejador;
9.     sigaction( SIGALRM, &sact, NULL );

10.     alarm(5);
11.     pause();
12.     exit(0);
13.   }
```

El sistema operativo cuenta con un planificador por prioridades. En el caso de procesos con la misma prioridad se utiliza un esquema *round robin*. Se considera que el proceso con un valor más alto de prioridad se ejecuta con anterioridad.

Se supone que el Proceso 1 sólo se va a activar cuando le llegue la petición de impresión del Proceso 4. El Proceso 1 está bloqueado en la lectura de una tubería con nombre esperando el aviso por parte de cualquier proceso que vaya a imprimir. El

Proceso 4 escribe en un directorio el trabajo de impresión y a continuación se comunica con el Proceso 1 a través de la tubería con nombre.

Considerando que el sistema operativo sigue un modelo de procesos (núcleo con ejecución dentro de los procesos de usuario) y núcleo expulsivo (permite el anidamiento de llamadas al sistema), se pide:

1. **[0,5 puntos]** Clasificar todos los procesos, incluido el proceso nulo, como procesos de usuario o procesos de núcleo. Justificar la respuesta.
2. **[2,5 puntos]** En la situación descrita, plantear un escenario en el que entre a ejecutar el proceso nulo y no haya finalizado ninguno de los procesos. Para ello:
    a. Describir las acciones que habrían llevado a cabo todos los procesos hasta ese mismo instante.
    b. Especificar en qué rutina de tratamiento de evento estaría cada uno de los procesos del escenario.
    c. El proceso nulo, ¿en qué modo de ejecución estaría? Justificar la respuesta.
3. **[2 puntos]** Supóngase que el proceso 2 está ejecutando la línea 5, el proceso 3 está bloqueado en la línea 11 y le quedan 4 segundos para que se active la señal SIGALRM, el proceso 4 está listo para ejecutar y el proceso 1 está bloqueado, en espera de ser despertado para llevar a cabo la tarea correspondiente. A partir de ese instante y hasta que mueran los procesos 2, 3 y 4, describir en orden los eventos que se generan en la ejecución de todos los procesos y cómo se lleva a cabo la planificación de los procesos. Para cada evento que se genere, especificar en qué estado se encuentran todos los procesos y en el contexto de qué proceso se lleva a cabo el tratamiento de dicho evento. Recuerde tratar la ocurrencia de la interrupción de reloj, estableciendo valores razonables para todos los parámetros que sean necesarios para resolver este apartado.
4. **[2 puntos]** Llevar a cabo el apartado anterior considerando un núcleo no expulsivo (no permite el anidamiento de llamadas al sistema). ¿Cuáles son las diferencias más importantes entre los núcleos expulsivos y no expulsivos?
5. **[0,5 puntos]** Si las prioridades de los procesos 1, 3 y 4 cambian y toman los valores 3, 4 y 5 respectivamente. ¿Qué situación anómala respecto a las prioridades podría darse en la ejecución concurrente de todos los procesos?
6. **[2 puntos]** Si en lugar de utilizar un modelo de procesos (núcleo con ejecución dentro de los procesos de usuario) se utilizara un modelo de interrupciones (núcleo con ejecución independiente), responder al apartado 3, destacando las principales diferencias entre estos dos modelos.
7. **[0,5 puntos]** ¿Cuál de los dos modelos anteriores elegiría para diseñar un sistema operativo monousuario, multiproceso, con arquitectura *microkernel* y que esté restringido al uso de pocos periféricos? Justificar la respuesta.

SOLUCIÓN.-

1. El proceso 1, proceso de *spooling*, es un proceso de *super-usuario* que proporciona servicio para la impresión. Por tanto, se trata de un proceso de usuario. Los procesos 2, 3 y 4 son también procesos de usuario así como también lo es el proceso creado por el proceso 2. El proceso nulo es un proceso de núcleo.
2. a) Para que pueda entrar a ejecutar el proceso nulo, el resto de procesos deben estar bloqueados, ya que el proceso nulo tiene la mínima prioridad y sólo consigue el procesador en esta situación. Un posible escenario sería el siguiente:
    - El proceso 1 está bloqueado en la lectura de una tubería con nombre a la espera de una petición de impresión, en este caso, por parte del proceso 4.
    - El proceso 2 ha creado al proceso 2 hijo. Éste se ha quedado bloqueado en la lectura del descriptor fd y el proceso padre está bloqueado esperando por la finalización del proceso hijo.
    - El proceso 3 ha armado la señal SIGALRM y se encuentra bloqueado esperando a que pasen los 5 segundos en la llamada `pause`.
    - Finalmente el proceso 4 se encuentra bloqueado en la escritura del trabajo de impresión en el directorio correspondiente.

    En esta situación, todos los procesos se encuentran bloqueados y entra a ejecutar el proceso nulo.

    b) Las rutinas de tratamiento en las que se encuentra cada uno de los procesos son:
    - El proceso 1 está bloqueado en la rutina de la llamada `read`.
    - El proceso 2 hijo está bloqueado en la rutina de la llamada `read`.
    - El proceso 2 padre está bloqueado en la rutina de la llamada `wait`.
    - El proceso 3 está bloqueado en la rutina de la llamada `pause`.
    - Finalmente, el proceso 4 está bloqueado en la rutina de la llamada `write`.

    c) El proceso nulo ejecuta en modo de ejecución sistema o núcleo, dado que se trata de un proceso de núcleo que ejecuta código del sistema operativo en modo sistema.
3. Suponemos que se generan interrupciones de reloj cada 10 ms. A la hora de responder a este apartado, sólo se van a considerar aquellas interrupciones de reloj que impliquen un cambio en el estado de los procesos. El proceso 2 ejecuta la línea 5. Ya ha salido del bloqueo de la llamada `wait`, por tanto, el proceso 2 hijo ya ha finalizado. En ese instante, el proceso 1 está bloqueado esperando la petición del proceso 4, el proceso 3 se encuentra bloqueado en la llamada `pause` y el proceso 4 está listo para ejecutar. La función `fprintf` es una función de biblioteca, que internamente requiere una llamada `write`. Uno de los argumentos de la llamada (4/0) genera una excepción que requiere ser tratada. El tratamiento de la excepción se realiza en modo sistema e implica que el proceso 2 finaliza. A continuación y dentro del contexto del proceso 2 se elige a un proceso para su ejecución. El único proceso elegible es el proceso 4. Se supone que la rodaja de tiempo para el proceso 2 ha permitido llevar a cabo todas estas acciones. El proceso 4 escribe el trabajo de impresión en el directorio, lo que implica una llamada

`write`, que cambia el proceso a modo sistema e implica un bloqueo a dicho proceso. El resto de procesos existentes, proceso 1 y proceso 3 se encuentran bloqueados. Por tanto, dentro del contexto de ejecución del proceso 4 se elige al único proceso planificable, es decir, el proceso nulo. Este proceso se encontrará en ejecución hasta que finalice las correspondientes operaciones de E/S necesarias para escribir el trabajo de impresión en el directorio. En ese periodo de tiempo se estarán intercalando los procesos nulo y 4 en la ejecución. Cuando haya finalizado la escritura del trabajo, el proceso 4 se pondrá en ejecución para comunicarse con el proceso 1 a través de una tubería con nombre. Realizará una invocación a la llamada `write`, que desbloqueará al proceso 1 que se encuentra bloqueado en la llamada `read`. Al tratarse de un proceso más prioritario, se llevará a cabo un cambio de contexto involuntario dentro del contexto del proceso 4, que provocará que el proceso 4 pase a estar en ejecución. El proceso 3 sigue bloqueado. El proceso 1 leerá de la tubería de impresión y procederá a llevar a cabo la operación de impresión, que implicará un bloqueo cuando realice la operación de E/S correspondiente. En ese momento se llevará a cabo un cambio de contexto voluntario al proceso 4 dentro del contexto del proceso 1, que hará que el proceso 4 pase a estar en ejecución. El proceso finaliza y dentro del contexto del mismo, se procederá a planificar el siguiente proceso, que dependiendo de si ha finalizado o no la operación de E/S del proceso 1, pondrá en ejecución al proceso 1 o al proceso nulo. En este momento se intercalará la ejecución del proceso 1 y nulo, dependiendo de las operaciones de E/S necesarias. Finalmente, el proceso 1 se quedará bloqueado en la lectura de la tubería, esperando por otros trabajos de impresión, dando paso al proceso nulo, hasta el momento en que se genere la señal SIGALRM, que permitirá desbloquear al proceso 3 y ponerle en ejecución. Ese proceso finaliza a continuación, quedando sólo el proceso 1 bloqueado y el proceso nulo en ejecución.

4. Dado que en el escenario anterior, sólo se da un cambio de contexto involuntario dentro del contexto del proceso 4, la diferencia sería que el proceso 4 finalizaría la llamada al sistema antes de dar paso a la ejecución del proceso 1. La diferencia entre los 2 núcleo es que en un núcleo expulsivo se producen el cambio de contexto involuntario antes de que finalice la llamada al sistema, lo que permite anidar determinadas llamadas al sistema y lo que implica problemas de sincronización más acuciantes que en el caso de núcleos no expulsivos.
5. En este caso se podría dar la situación de inversión de prioridades, dado que un proceso de mayor prioridad espera a que finalice un proceso de menor prioridad por su dependencia respecto a un tercer proceso.
6. La diferencia entre estos 2 modelos es que en el modelo de interrupciones, el sistema operativo tiene su propia región de memoria y su propia pila de sistema y ejecuta fuera de todo proceso, dado que tiene su propio contexto independiente de los contextos de los procesos. La diferencia por tanto radica en el contexto en el que se lleva el tratamiento de los eventos, que se realiza dentro del contexto del sistema operativo.
7. Para sistemas operativos sencillos con arquitectura *microkernel* puede ser más apropiado utilizar un modelo de interrupciones, con la ejecución del código del sistema operativo fuera del contexto de todo proceso.