

Grupo: NIA:

Nombre y apellidos:

PROBLEMAS

INSTRUCCIONES A SEGUIR:

- Es obligatorio seguir todas las siguientes instrucciones de realización del examen.
- Recuerde que ha de responder a lo pedido, de forma correcta, en el tiempo dado y en el espacio indicado.
- **No** desgrape el examen.
- Es obligatorio la entrega del examen.

- Indique su nombre, NIA y grupo en la primera hoja.
- Será necesario identificarse (DNI o carnet universitario) en todo momento.

- **No** se pueden utilizar: libros, apuntes, calculadora, móvil (o similares dispositivos).
- Todo aparato electrónico debe estar apagado y fuera del alcance del estudiante.
- Los apuntes y libros deben estar en la mochila y fuera de alcance del estudiante.
- No se permiten estuches ni similares encima de la mesa.

- Para la realización de esta parte del presente examen se dispondrá de **120 minutos**.
- No se volverá a dar hoja de respuesta si no es por causa justificada.

- Aunque las pruebas evaluables pueden calificarse de forma acumulativa (puntuar por apartados), recuerde que un error grave puede suponer un cero en su valoración total.
- Por favor revise la respuesta para comprobar que realiza la funcionalidad pedida por completo para optar a la máxima calificación en el ejercicio.
- El planteamiento solo de un problema no se tomará como solución válida.

Grupo: NIA:

Nombre y apellidos:

Ejercicio 2 (3,5 puntos)

En la empresa en la que estamos trabajando disponemos de un sistema operativo básico que usa un *kernel* monolítico con un planificador *Round-Robin* (RR) para todos los procesos (de forma similar al introducido en la asignatura). El sistema también dispone de un driver de teclado que usa interrupciones.

Esta empresa quiere que añadamos la siguiente funcionalidad: poder habilitar el cambio de la planificación usada de *Round-Robin* (RR) a *First In First Out* (FIFO) y viceversa usando el teclado mediante una nueva llamada al sistema.

Si está habilitado el cambio por teclado, al pulsar la tecla 'f' del teclado se usará la planificación FIFO y si se pulsa la tecla 'r' se pasará a planificación RR. Esto puede afectar al proceso en ese momento en curso dado que el cambio se hace en el mismo instante de pulsar la tecla. En el caso de pasar a RR se comenzará con una rodaja entera.

Si no está habilitado el cambio por teclado, simplemente se usará RR y la pulsación de las teclas no tendrá el efecto anteriormente descrito.

La empresa nos da unos fragmentos de código en los que basarnos para trabajar:

Id. Frag.	Fragmento de Código
1	<ul style="list-style-type: none">• Proc = ObtenerPrimerProceso (Teclado.Bloqueados)• Si Proc == NULL<ul style="list-style-type: none">○ return• Proc->estado = LISTO• InsertarAlFinal(Listos, Proc)• cambioContexto(procesoAnterior, procesoActual)
2	<ul style="list-style-type: none">• procesoAnterior = procesoActual• procesoActual = planificador()• procesoActual->estado = EJECUTANDO• procesoActual->rodaja = RODAJA• cambioContexto(procesoAnterior, procesoActual)
3	<ul style="list-style-type: none">• Insertar_Interrupcion_Software(Despertar_bloqueados_teclado)• Generar_Interrupcion_Software()
4	<ul style="list-style-type: none">• Insertar_Interrupcion_Software(Tratar_rodaja_reloj)• Generar_Interrupcion_Software()

Grupo: NIA:

Nombre y apellidos:

Usando los fragmentos de código anteriores, **se pide** indicar el diseño e implementación en pseudocódigo de los siguientes cambios que hay que hacer en el sistema operativo inicial para añadir la funcionalidad pedida:

a) Disponer de una nueva llamada al sistema cuya declaración es:

```
void let_selection_from_keyboard ( int is_allow ) ;
```

que permite indicar si se permite seleccionar el planificador mediante teclado mediante un valor de `is_allow` mayor o igual a uno.

Si `is_allow` es un número negativo el *kernel* ignorará esta llamada.

Si `is_allow` es cero, se desactivará esta funcionalidad y se pasará a RR, hasta que con una nueva llamada con parámetro uno la vuelva a activar.

Por defecto la funcionalidad está desactivada.

b) Actualizar el tratamiento del reloj (interrupción hardware y software) de forma que si la variable global en el *kernel* `kern_sch_used` tiene un valor de RR entonces se realiza una planificación *Round-Robin* y si tiene un valor de FIFO realizará la planificación asociada. Inicialmente el valor de `kern_sch_used` es de RR.

c) Actualizar el tratamiento del teclado (interrupción hardware y software) de forma que además de funcionar como un teclado normal y habitual, si está activada la selección por teclado entonces compruebe si se pulsa las teclas 'r' o 'f' de forma que actualiza el valor de `kern_sch_used` de la forma deseada por la empresa.

Ha de contestar en el espacio dado para la respuesta indicando los fragmentos de código usado de la forma '<fragmento X>' siendo X el *Id. Frag.* asociado. Si fuera necesario añadir antes, después, antes y después líneas de código adicionales, indíquelo claramente usando una notación similar a la dada en los fragmentos de código.

NOTA:

Por favor revise la respuesta para comprobar que realiza la funcionalidad pedida dado que si falta cualquier línea que impida el correcto funcionamiento se dará como no válida la respuesta asociada a dicha línea de la funcionalidad pedida.

Grupo: NIA:

Nombre y apellidos:

Ejercicio 3 (2,5 puntos)

Disponemos de una maquina monoprocesador de 32 bits y queremos implementar un sistema de ficheros para un sistema operativo UNIX con una caché de bloques que permite leer (bread) y escribir (bwrite) bloques de disco.

El sistema de ficheros a implementar tiene la siguiente estructura en disco:

1 bloque	1 bloque	1 bloque	n bloques
Superbloque	Bloque con inodos	Bloque con nombres	Bloques de datos
0	1	2	3 N

Donde los requisitos del sistema de ficheros incluyen:

1. El tamaño de bloque es de 1024 bytes (1 KiB).
2. Hay 800 bloques en total en el sistema de ficheros.
3. Solo hay ficheros (que pertenecen a un directorio raíz común para todos), y como máximo son 50 ficheros.
4. El superbloque ocupará un bloque de disco y será el bloque inicial (bloque 0).
5. El superbloque contendrá la gestión de bloques libres/ocupados y de i-nodos libres y ocupados. Tiene un vector de 50 caracteres para los i-nodos y otro vector de 800 caracteres para los bloques, donde '0' indica que está libre y '1' que está ocupado.
6. Cada i-nodo ocupa 20 bytes
7. Cada i-nodo contiene, entre otras cosas, el tipo (regular, enlace simbólico, etc.) y un único bloque indirecto que referenciarán a los bloques de datos asociados a dicho fichero. Cada referencia a bloque ocupa 4 bytes.
8. El bloque de nombres contendrá un array de 50 posiciones donde cada una es una cadena de 20 caracteres que representa el nombre del fichero asociado.
9. Cada fichero solo puede ser abierto por un proceso, con un puntero de lectura y escritura particular para ese proceso (compartido por las aperturas de ese proceso).

Se pide:

- a) Complete el diseño de las estructuras dadas del sistema de ficheros que siempre estarán presentes en disco, indicando todos los campos que tendrá el superbloque y un i-nodo de este sistema de ficheros.
- b) Diseñar las estructuras en memoria que permitan trabajar con el sistema de ficheros anteriormente descrito.
- c) Pseudocódigo la función de tratamiento de bloques bmap asociado al diseño.
- d) Pseudocódigo las funciones de tratamiento de i-nodos: ialloc, ifree y namei.

Grupo: NIA:

Nombre y apellidos:

Ejercicio 4 (2,5 puntos)

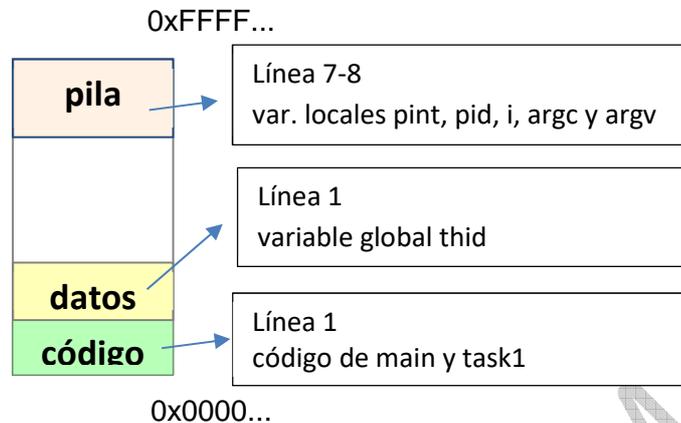
Se dispone del siguiente código fuente de un programa:

```
01 pthread_t thid;
02 void *task1 (void * arg) {
03     free(arg);
04     /* punto C */
05 }
06 main (int argc, char **argv) {
07     int *pint;
08     int pid, i;
09     /* punto A */
10     pint = malloc (100*sizeof(int));
11     pthread_create(&thid,NULL,task1,(void *)pint);
12     pid = fork();
13     /* punto B */
14     if (0 == pid) {
15         pthread_join(&thid,NULL);
16         free(pint);
17     } else {
18         waitpid(pid, NULL, 0);
19         /* punto D */
20     }
21 }
```

Grupo: NIA:

Nombre y apellidos:

Si en el punto A de ejecución, la imagen del proceso asociado se representa:



Se pide:

- Dibuje la imagen de los procesos (utilizando la representación dada) en el punto **B**, indicando para cada cambio producido en la imagen del proceso desde el punto anterior, la razón del cambio y en la línea de código en que se produce.
- Dibuje la imagen de los procesos (utilizando la representación dada) en el punto **C**, indicando para cada cambio producido en la imagen del proceso desde el punto anterior, la razón del cambio y en la línea de código en que se produce.
- Dibuje la imagen de los procesos (utilizando la representación dada) en el punto **D**, indicando para cada cambio producido en la imagen del proceso desde el punto anterior, la razón del cambio y en la línea de código en que se produce.
- ¿Considera que hay algún fallo en el uso de las llamadas de gestión de memoria en el fragmento de programa dado? Razone brevemente su respuesta.

NOTAS IMPORTANTES:

- Se ejecuta en un sistema operativo para **monoprocesador** de 32 bits con planificador **FIFO**.
- Los procesos ligeros son tratados de igual forma que los procesos pesados por el planificador.
- Los hilos son de tipo **PTHREAD_CREATE_JOINABLE** por defecto.
- La gestión de memoria **no** usa **COW (Copy-On-Write)**.
- Es absolutamente imprescindible que justifique en qué línea de código se produce un cambio en la imagen de memoria, qué cambio es y porqué se produce.
- La simple representación gráfica de la imagen de memoria no se considera respuesta válida.