



SISTEMAS OPERATIVOS. CUESTIONES
5 de Febrero 2013.

Nombre _____ DNI _____

Apellidos _____ Grupo _____

Cuestión 1. (0,75 puntos) Un sistema de ficheros UNIX utiliza bloques de 2048 bytes y direcciones de disco de 32 bits. Los nodos-i contienen 10 direcciones de disco para bloques de datos, una dirección de bloque índice indirecto simple y una dirección de bloque índice indirecto doble. ¿Cuál es el tamaño máximo de un fichero en este sistema?

Cuestión 2. (0,75 puntos) Un disco tiene las siguientes peticiones pendientes: 98, 183, 37, 122, 14, 124, 65, 57. La cabeza de lectura/escritura está inicialmente sobre el cilindro 53 y su movimiento es hacia cilindros crecientes. Indique el orden de atención de dichas solicitudes ante los siguientes algoritmos:

	1	2	3	4	5	6	7	8
FCFS								
SCAN								
C-SCAN								

Cuestión 3. (0,75 puntos) Un dispositivo de tipo carácter implementa la siguiente función:

```
static ssize_t device_read(struct file *filp, char *buffer, size_t length,
loff_t * offset);
```

Ponga un ejemplo de orden Linux en que se ejecutaría esta función. ¿Qué debe almacenar la cadena buffer? ¿Qué sentido tiene utilizar la función "copy_to_user(buffer, msg_Ptr, BUFFER_LEN)" dentro de device_read? *Nota: copy_to_user es análoga a copy_from_user.*

Cuestión 4. (0,75 puntos) Considere el siguiente código:

```
int a_global=0;
int main() {
    pid_t pid;
    int b_local = 0;
    int *p_heap;
    int i;

    p_heap = (int*) malloc (sizeof(int));
    *p_heap = 0;

    for (i=0;i<3;i++) {
        if ( (pid=fork()) == 0) {
            a_global+=2; b_local+=2; (*p_heap) +=2;
            printf("Child. pid=%u ppid=%u. i=%d a=%d b=%d\n",getpid(),getppid(),i,a_global,b_local,*p_heap);
        }
        else {
            a_global++; b_local++; (*p_heap)++;
            printf("Parent. pid=%u ppid=%u. i=%d a=%d b=%d\n",getpid(),getppid(),i,a_global,b_local,*p_heap);
        }
    }
}
```

Asumiendo que un proceso padre siempre es más prioritario que sus hijos, que el proceso original tiene PID 100 (y es hijo de *Init*) y que los PID se van asignando de forma consecutiva, indique qué se mostrará por pantalla al ejecutar el código.

Cuestión 5. (0,75 puntos) Considere el siguiente código:

```
char buf1[4] = "XXXX";
char buf2[4] = "AAAA";
int main() {
    int fd1,fd2;
    fd1=open("prueba",O_RDWR | O_CREAT | O_TRUNC, 0666 );

    if ( fork() == 0 ) {
        fd2=open("prueba",O_RDWR );
        write(fd1,buf1,4);
        write(fd2,buf2,4);
        close(fd2);
        close(fd1);
    }
    else {
        write(fd1,buf1,4);
        wait(NULL);
        read(fd1,buf2,4);
        close(fd1);
    }
}
```

Indique si la ejecución del código generará algún error. En cualquier caso, ¿cuál será el contenido del fichero *prueba* al finalizar la ejecución? **Justifique la respuesta.**

Cuestión 6. (0,75 puntos) Implemente un semáforo general a partir de cerrojos y variables condicionales. El comportamiento deseado del semáforo general sería:


```
sem_wait(mi_sem_t *s) {
    s->valor = s->valor -1;
    if (s->valor <0) {
        <Bloquear>
    }
}

sem_post(mi_sem_t *s) {
    s->valor = s->valor + 1;
    if (s->valor <= 0) {
        <Desbloquear un proceso >
    }
}
```

Especifique el tipo de datos *mi_sem_t* y proponga la implementación de *sem_wait* y *sem_post*.

Cuestión 7. (0,75 puntos) Indique qué secciones de un ejecutable (ELF) pasan a ser regiones de memoria de un proceso que inicialice su mapa de memoria con ese ELF. Además, indique al menos otras cuatro posibles regiones de memoria de un proceso que no procedan directamente del ejecutable.

Cuestión 8. (0,75 puntos) Explique la traducción de la dirección virtual a dirección física en un sistema con memoria virtual paginada y TLB, indicando a qué estructuras se accede en cada paso. Indique en qué fases de la traducción debe intervenir el sistema operativo (indicando cómo/por qué empieza a ejecutarse código del SO) y en cuáles sólo interviene el hardware.

	SISTEMAS OPERATIVOS. PROBLEMAS 5 de Febrero 2013.
	Nombre _____ DNI _____ Apellidos _____ Grupo _____

Problema 1. (2 puntos) Un sistema de ficheros basado en i-nodos y mapa de bits contiene la siguiente información:

Mapa de bits: 10111101110....0

Nodos-i:

Info/Nodo-i	N2	N3	N4	N5	N6	N7
Tamaño	1		1	2	1	1
#Enlaces	NA	NA	NA	1		NA
Tipo F/D	D	D	D	F	F	D
Directo		2	3	4	9	8
Indirecto	-	-	-	5	-	-

Bloques:

B0		B2		B3		B4	B5	B7	B8		B9
.	2	.	3	.	4			Datos	.	7	Datos
..		..	2	..	2				..	4	
A	3	D	6	E	6						
B	4			F	7						
C	5										

Se pide:

- Rellene los huecos para que el sistema sea consistente. Asuma para ello que el tamaño se expresa en bloques.
- Dibuje el árbol del directorio empleando óvalos para los directorios, rectángulos para los ficheros y triángulos para los datos.

Problema 2. (2 puntos) En un comedor escolar hay 2 cocineros y n alumnos. Cada alumno debe coger de la mesa un bocadillo y una bebida. Uno de los cocineros repone los bocadillos y otro las bebidas. El funcionamiento del comedor es el siguiente:

Los alumnos pasan de uno en uno: no llega otro alumno a la mesa hasta que sale el anterior. Cuando llega un alumno coge primero un bocadillo, luego una bebida, después se va de la mesa y come. Si no hubiera alguna de las dos cosas avisa al cocinero correspondiente y espera a que repongan.

El cocinero encargado de los bocadillos espera a que le llamen y cuando un alumno le llama repone p bocadillos.

El cocinero encargado de las bebidas espera a que le llamen y cuando un alumno le llama repone b bebidas.

Escriba un programa que sincronice a los alumnos y cocineros de acuerdo con la estructura siguiente:

```
Bocadillos() {  
    while (TRUE) {  
        //Espera a que un alumno le avise  
  
        //Repone p bocadillos  
  
        // Avisa a alumnos  
    }  
}  
Bebidas() {  
    while (TRUE) {  
        //Espera a que un alumno le avise  
  
        //Repone b bebidas  
  
        // Avisa a alumnos  
    }  
}
```

```
void Alumno() {  
    while (TRUE) {  
        // Si no hay bocadillos, avisa a  
        cocinero (y espera)  
  
        // Coge un bocadillo  
  
        // Si no hay bebidas, avisa a  
        cocinero (y espera)  
  
        // Coge una bebida  
  
        comer()  
    }  
}
```