Apellidos: _____ Nombre: _____ DNI: _____

Sistemas Operativos – Grado Ingeniera Informática UDC. Julio 2019

Sólo puede usar lápiz, bolígrafo y calculadora.

Tiempo máximo para todo el examen: 3h

Parte Sistema Ficheros

(Se deben contestar correctamente todas las cuestiones de cada pregunta para puntuar la misma).

Puntuación preguntas: P1: 0.7, P2: 0.4, P3: 0.5, P4: 0.4

P1) Un sistema de archivos tipo UNIX System V tiene un tamaño de bloque de 2Kbytes, i-nodos con 10 direcciones directas, una indirecta simple, una indirecta doble y una indirecta triple. Utiliza direcciones de bloque de 8 bytes. Calcular el tamaño máximo de un fichero al utilizar los niveles de indirección simple, doble y triple.

Tamaño máximo usando puntero de indirección simple: 0.5 Mbytes + 20 Kbytes

Tamaño máximo usando puntero de indirección doble: 128.5 Mbytes + 20 Kbytes

Tamaño máximo usando puntero de indirección triple: 32 Gbytes + 128.5 Mbytes + 20 Kbytes

P2) En ese sistema de archivos UNIX (tamaño de bloque de 2Kbytes), el *boot* ocupa los 2 primeros bloques de su partición de disco. Por tanto, el superbloque comienza en el bloque lógico 2 de la partición de disco del SF (se numera a partir del bloque 0). El fichero "datos" tiene asociado el *inodo* 325 y ocupa el bloque (lógico) 30 desde el principio de la partición. El tamaño del *inodo* es de 128 bytes. Calcula el tamaño (en Kbytes) del *superbloque*.

Tamaño del superbloque : 16 Kbytes

P3) En ese sistema de archivos (pregunta P1), un proceso abre el archivo "datos" (tamaño 200 Mbytes) en modo lectura/escritura, *open* ("datos", *O_RDWR*), que se encuentra en el directorio de trabajo del proceso (la apertura no retorna ningún error). Consideremos el siguiente trozo de código en esa apertura:

```
char c;
int fd=open("datos", O_RDWR); /* es la primera apertura de fichero por
                               parte del proceso */
lseek(fd, 8388608, SEEK_SET); /* SEEK_SET indica que el desplazamiento
                               se considera a partir del origen del fichero
                               (8388608= 223) */
c=fgetc(fd);
printf ("La apertura del fichero retorna descriptor %d\n", fd);
close(fd);
```

Apellidos: _____ Nombre: _____ DNI: _____

Calcular cuántos bloques tendría el SO que leer del disco, discriminando entre bloques de datos y de índices, para realizar la operación anterior:

$c=fgetc(fd);$

suponiendo el buffer caché vacío. (Nota: $8388608= 2^{23}$)

A. Número de bloques de índices que es necesario leer: 2

B. Número de bloques de datos que es necesario leer: 1

C. Marcar las opciones que son ciertas:

Al ejecutar *printf* se incrementa solo el tiempo de usuario del proceso

Al ejecutar *printf* se incrementa solo el tiempo de sistema del proceso

printf es una llamada al sistema operativo

lseek es una llamada al sistema operativo

P4) Supongamos las siguientes secuencias de comandos:

1. Se crean 2 *hard link* al fichero "datos" (cuyo número de inodo es 325, tamaño 200 Mbytes y num. de *hard links*=1) en su mismo directorio:

```
In datos datos1
```

```
In datos datos2
```

2. Posteriormente se crea un *soft link* al fichero *datos1*:

```
In -s datos1 slink1 /* crea soft link slink1 */ /* el comando ls -l mostraría slink1 -> datos1 */  
y otro un soft link al fichero datos2:
```

```
In -s datos2 slink2 /* crea soft link slink2 */ /* el comando ls -l mostraría slink2 -> datos2 */
```

3. Finalmente, se crea un *hard link* al fichero *slink2*:

```
In slink2 hlink2
```

Contestar a lo siguiente:

A. Indicar el tamaño (en Gbytes, Mbytes o bytes) del fichero *hlink2*: 6 bytes

B. Indicar cuál es el número de (*hard*) *links* del fichero *slink2*: 2

C. Una vez creados los ficheros *slink1* y *slink2*, al borrar el fichero "datos" (*rm datos*) se decrementa su número de *hard links*. ¿Se puede acceder al contenido del fichero con número de inodo 325 a través del link simbólico *slink2*? Sí

D. Tras la operación *rm datos*, ¿cuál es el número de *hard links* de *slink1*? 1

Apellidos: _____ Nombre: _____ DNI: _____

Sistemas Operativos - Grado Ingeniera Informática UDC. Julio 2019

Sólo puede usar lápiz, bolígrafo y calculadora.

Tiempo máximo para todo el examen: **2h45m**

Parte Memoria.

1. Un proceso produce la siguiente cadena de referencias a memoria. Ilustrar el funcionamiento de los siguientes algoritmos de reemplazo si el proceso tiene asignación fija de 3 frames de memoria. En la primera fila de la tabla aparece la cadena de referencias y en la última hay que indicar los fallos de página. Cada fila indica una frame asignada al proceso y para puntuar cada apartado tiene que ser correcto el número de fallos y la asignación de páginas lógicas a frames.

FIFO, num. fallos de página: 11 (0.15 puntos)

A	B	C	D	A	B	E	A	B	C	D	E	B	A	B
A	A	A	D	D	D	E	E	E	E	E	E	B	B	B
	B	B	B	A	A	A	A	A	C	C	C	C	A	A
		C	C	C	B	B	B	B	B	D	D	D	D	D
F	F	F	F	F	F	F			F	F		F	F	

LRU, num. fallos de página: 12 (0.15 puntos)

A	B	C	D	A	B	E	A	B	C	D	E	B	A	B
A	A	A	D	D	D	E	E	E	C	C	C	B	B	B
	B	B	B	A	A	A	A	A	A	D	D	D	A	A
		C	C	C	B	B	B	B	B	B	E	E	E	E
F	F	F	F	F	F	F			F	F	F	F	F	

FIFO 2ª oportunidad, num. fallos de página: 11 (0.1 puntos)

A	B	C	D	A	B	E	A	B	C	D	E	B	A	B
A*	A*	>A*	D*	D*	>D*	E*	E*	E*	E	>E	>E*	B*	B*	B*
>	B*	B*	>B	A*	A*	>A	>A*	>A*	C*	C*	C*	>C	A*	A*
	>	C*	C	>C	B*	B	B	B*	>B	D*	D*	D	>D	>D
F	F	F	F	F	F	F			F	F		F	F	

-(0.05 puntos) De los algoritmos anteriores, ¿cuáles se pueden usar en un escenario de asignación variable de memoria a los procesos y reemplazo global?

Los tres algoritmos

-(0.05 puntos) De los algoritmos anteriores, ¿cuáles se pueden usar en un escenario de asignación fija de memoria a los procesos y reemplazo global?

Ese escenario no es posible

2. (0.6 puntos) Con un tamaño de página de 1KB (1 Kbyte), y entradas de tabla de página de 4 bytes, con un sistema de paginación en varios niveles, ¿cuántos niveles de tablas de páginas son necesarios en una arquitectura de direcciones virtuales de 34 bits si cada tabla de página ocupa una página? Explique los cálculos con precisión.

Numero de niveles: 3

1KB = 2^{10} bytes; $2^{10}/2^2 = 2^8$ entradas en cada tabla de páginas que ocupa una página y además se necesitan 8 bits para direccionar cada nivel

con un nivel = $2^8 \times 2^{10} = 2^{18}$ bytes

con dos niveles = $2^8 \times 2^8 \times 2^{10} = 2^{26}$ bytes

con tres niveles = $2^8 \times 2^8 \times 2^8 \times 2^{10} = 2^{34}$ bytes

-¿Cuál es el formato de las direcciones virtuales?

8 bits para la entrada en la TP de primer nivel, 8 bits para la entrada en la TP de segundo nivel, 8 bits para la entrada en la TP de tercer nivel, 10 bits para el offset en la página

-¿Sería posible con esta arquitectura direccionar un espacio de direcciones físicas de 8TB (8 Terabytes)? 1TB = 2^{40} bytes. Si/No ¿Por qué?

8TB = 2^{43} bytes

Como las páginas son de 1KB, se necesitan 2^{33} páginas de 1KB. Las entradas de las TP son de 32 bits por lo que pueden direccionar 2^{32} páginas de 1KB, es decir 2^{42} bytes, por tanto no se pueden direccionar un espacio de direcciones físicas de 8 Terabytes. Realmente en la práctica no se pueden utilizar los 32 bits para direccionar páginas porque algunos bits son para permisos o bit de presencia si tuviésemos memoria virtual, pero incluso en ese caso extremo de usar los 32 bits para direccionar, no se podrían alcanzar los 8 Terabytes.

En las preguntas 3, 4, 5 sólo deber contestar Verdadero/Falso (o no contestar) pero cada respuesta incorrecta o ambigua invalida una correcta dentro de la pregunta.

3. (0.3 puntos) En una arquitectura de memoria con un registro base y un registro límite

-puede haber varios procesos activos en memoria: V

-permite swapping (intercambio de procesos): V

-un proceso que necesita 16Mbytes de memoria puede correr en una máquina con 8Megabytes de RAM: F

4. (0.3 puntos) En una entrada de TLB está contenida esta información

-num página lógica: V

-num página física: V

-offset en la página lógica: F

-offset en la página física: F

-información de permisos de la página: V

-dirección base de la tabla de páginas raíz o de primer nivel: F

5. (0.3 puntos) Al incrementar el tamaño de página

-se reduce el tamaño de la tabla de páginas: V

-se incrementa el tiempo de servicio de un fallo pagina: ___V_____

-se reduce el numero de fallos en la TLB: ___V_____

SISTEMAS OPERATIVOS

Segundo curso Grado en Informática. Julio 2019

APELLIDOS, Nombre:.....

Tiempo: 2h 45m

1	2	T
0.75	1.25	2

1. Considérese el siguiente código en C (se supone que contiene todos los ficheros include necesarios y que compila correctamente)

```
main(int argc, char * argv[])
{
    int i=100;

    pid=fork()*fork();  /******/

    if (pid>0)
        execl("./a.out", "./a.out",NULL);

    printf ("i vale: %d\n",i);
}
```

donde el código de ./a.out es el siguiente: (tambien se supone contiene todos los ficheros include necesarios y que compila correctamente)

```
int i;
main (int argc, char *argv[])
{
    i++;
    printf ("el valor de i es: %d\n",i);
}
```

- a) ¿Qué salida produce dicho código? (suponemos que tanto *fork()* como *execl()* se completan correctamente)
- el valor de i es: 1*
i vale: 100
i vale: 100
i vale: 100
- b) ¿Qué salida produciría si una de las llamadas *fork()* produce un error (*execl()* se completa correctamente)?
- i vale: 100*
i vale: 100
- c) ¿Qué salida produciría si las dos llamadas *fork()* producen un error (*execl()* se completa correctamente)?
- el valor de i es: 1*

2. Un sistema monoprocesador con multiples colas tiene dos colas de planificación: SYS para los procesos del sistema y USR para los procesos de los usuarios. En el cuadro se muestra la planificación del sistema con cuatro procesos A, B, C y D. CPU indica que proceso está en CPU; SYS y USR los contenidos de las colas SYS y USR y E/S qué procesos están realizando una entrada/salida.

CPU	B	B	C	C	C	A	D	D	B	D	C	C	C	C	C	C	D	D	A	A	D	D	A
SYS		C																					
USR	A	A D	A D	A D	A D	D			D			D	D A	D A	D A	D A	A	A	D	D	A	A	
E/S			B	B	B	B C	B A	B A	C A	C A	D A												

- a) Asumiendo que el instante en que entra el proceso B en CPU es el instante 0, rellenar el siguiente cuadro donde indicaremos, para cada proceso, su duración en el formato x-(y)-z (que indica una ráfaga de CPU de duración x, seguida de una ráfaga de e/s de duración y, seguida de una ráfaga de CPU de duración z), T_A , su instante de llegada, T_S su tiempo de servicio, T_R su tiempo de retorno, T_e su tiempo de espera y si es un proceso de usuario o del sistema (SYS/USR)

	Ráfagas	T_A	T_S	T_R	T_e	SYS/USR
Proceso A	1-(6)-3	0	10	23	13	SYS
Proceso B	2-(6)-1	0	9	9	0	USR
Proceso C	3-(5)-6	1	14	15	1	USR
Proceso D	3-(1)-4	1	9	21	13	SYS

- b) ¿Qué planificación existe entre las colas SYS y USR?
Prioridades apropiativas. La cola SYS tiene mayor prioridad
- c) de los siguientes algoritmos: RRq1 (RoundRobin quanto 1), RRq2, FCFS, SJF, SRTF. ¿cuales pueden haber sido usados para planificar la cola SYS?
RRq2, FCFS, SJF, SRTF
- d) de los siguientes algoritmos: RRq1 (RoundRobin quanto 1), RRq2, FCFS, SJF, SRTF. ¿cuales pueden haber sido usados para planificar la cola USR?
RRq2

Apellidos: Nombre: DNI:

Sistemas Operativos – Grado Ingeniería Informática UDC. Julio 2019

Sólo puede usar lápiz, bolígrafo y calculadora.

Tiempo máximo para todo el examen: **2h45m**

Parte E-S [1.50 puntos]

1. **(0.50 puntos)** Indíquese qué sucede al ejecutar el siguiente programa. Se asume que el fichero file1.dat existe en el directorio actual y el comando `cat file1.dat` nos muestra que su contenido es 0123456789. Además el fichero file2.dat no existe actualmente.

```
1. 1: #include "includes.h"           // STDIN_FILENO = 0; STDOUT_FILENO = 1;
1. 2 int main() {
1. 3   char buf[20]; int i=0,ifd,ofd,bk;
1. 4   ifd = open("file1.dat",O_RDONLY);
1. 5   ofd = open("file2.dat",O_WRONLY|O_CREAT, 0666);
1. 6   bk=dup(STDOUT_FILENO);
1. 7   close(STDOUT_FILENO); close(STDIN_FILENO);
1. 8   dup(ifd); dup(ofd);
1. 9   while (read(STDIN_FILENO,buf+i,1)) i++;
1.10  write(STDOUT_FILENO,buf,i);
1.11  close(STDOUT_FILENO);
1.12  dup(bk);
1.13  write(STDOUT_FILENO,"done\n",5);
1.14  close(bk); close(ifd); close(ofd);
1.15 } //read() devuelve 0 cuando no hay datos que leer
```

Asumiendo que el contenido de la tabla de ficheros abiertos del proceso al ejecutar las líneas 1.2 y 1.5 fuese el que se indica a continuación, indíquese cuál es el contenido de las entradas 0 a 6 de dicha tabla, en la línea 1.13.

estado 1.2		estado 1.5		estado 1.13	
0	teclado-in	0	teclado-in	0	file1.dat
1	pantalla-out	1	pantalla-out	1	pantalla-out
2	pantalla-err	2	pantalla-err	2	pantalla-err
3		3	file1.dat	3	file1.dat
4		4	file2.dat	4	file2.dat
5		5		5	pantalla-out
6		6		6	

2. **(0.50 puntos)** Disponemos de un disco con dos platos y dos caras por plato que contiene 1024 cilindros, y 256 sectores por pista. Cada sector contiene 512bytes.

¿Cuántos sectores hay en total? ¿cuántos bloques ve el sistema operativo si lo formateamos con bloques de 1024 bytes? Razónese la respuesta (muestre el resultado como una potencia de 2).

El número de sectores es: 2^{20} . El número de bloques es: 2^{19}
Justificación: 1024 cilindros, 2 caras, 2 platos $\rightarrow$ 4096 pistas = 2^{12} pistas. 2^{12} pistas x 256 sectores/pista = 2^{20} sectores. 2^{20} sectores x 1 bloques/2sectores = 2^{19} bloques.

3. (0.50 puntos) Indique V/F (Verdadero/Falso) en las siguientes sentencias¹:

F El registro de estado de un controlador de dispositivo permite enviar comandos para indicar al dispositivo qué operación debe realizar.

F El device-driver es la capa del software de Entrada/Salida que chequea si un determinado bloque está en la caché de bloques.

V Un controlador DMA configurado en *modo ráfaga* (*Burst Mode DMA*) tiene mayor prioridad de acceso al bus que si usase *modo por robo de ciclos* (*Cycle-stealing DMA*).

F Los *minor numbers* de los dispositivos de disco `/dev/sda` y `/dev/sdb1` son idénticos.

F `int aio_read (struct aiocb *aiocbp)` devuelve el número de bytes leídos en una lectura asíncrona.

¹OJO: La respuesta debe incluirse en el cuadro a la izquierda de cada sentencia. Cada respuesta correcta suma 0.10, cada respuesta incorrecta resta 0.10. Déjese en blanco el cuadro, o táchese completamente, para anular una respuesta.