Apellidos:_____ Nombre:_____ DNI: _____

Sistemas Operativos – Grado Ingeniería Informática UDC. Junio 2022
Sólo puede usar lápiz, bolígrafo y calculadora.

Parte **Sistema Ficheros (**2 puntos)

Examen sin identificación completa NO se califica.
Esta parte NO admite entrega de hojas adicionales. Hay que contestar en las páginas y espacios proporcionados.

Puntuación preguntas: P1: 0.6, P2: 0.5, P3: 0.3, P4:0.6

P1) Un sistema de archivos tipo UNIX tiene un tamaño de bloque de 8Kbytes, i-nodos con 10 direcciones directas, una indirecta simple, una indirecta doble y una indirecta triple. Utiliza direcciones de bloque de 8 bytes. Calcular cuántos bloques de disco son necesarios (en el área de datos) para representar el archivo *"datos"* cuando su tamaño es de 8 Gbytes + 32 Mbytes + 8 Kbytes. Discriminar cuántos bloques son de datos y cuántos de índices.

Un sistema de ficheiros tipo UNIX ten un tamaño de bloque de 8Kbytes, i-nodos con 10 enderezos directos, un indirecto simple, un indirecto dobre e un indirecto triple. Usa enderezos de bloque de 8 bytes. Calcular cantos bloques de disco son necesarios (na área de datos) para representar o ficheiro *"datos"* cando o seu tamaño é de 8 Gbytes + 32 Mbytes + 8 Kbytes. Discriminar cantos bloques son de datos e cantos de índices.

A UNIX file system has a block size of 8Kbytes, i-nodes with 10 direct addresses, a single indirect, a double indirect and a triple indirect address. It uses 8-byte block addresses. Calculate how many blocks are needed (in the data area) to represent the file *"datos"* when its size is 8 Gbytes + 32 Mbytes + 8 Kbytes. Discriminate how many blocks are data blocks and how many index blocks.

Nº bloques de datos/Nº bloques de datos/Number of data blocks: **1M + 4K + 1** (0.30p)

Nº de bloques de índices/Nº de bloques de índices/Number of index blocks: **1031** (0.25p)

Fragmentación interna del fichero/Fragmentación interna do ficheiro/Internal file fragmentation: **0** (bytes) (0.05p)

P2) Un proceso abre el archivo anterior (en P1) *"/home/usr1/datos" (tamaño* 8 Gbytes + 32 Mbytes + 8 Kbytes*)* en modo lectura, cuyo número de *hard links* es 1. El usuario efectivo del proceso coincide con el propietario del fichero, y tiene además los permisos de acceso a los directorios *raíz*, *home* y *usr1*. Las cachés de datos e inodos están inicialmente vacías y la entrada *usr1* está en el séptimo bloque del directorio *home*, mientras las demás entradas están en el primer bloque de sus directorios padre. Indicar lo siguiente referente al trozo de código:

Un proceso abre o ficheiro anterior (en P1) *"/home/usr1/datos" (tamaño* 8 Gbytes + 32 Mbytes + 8 Kbytes*)* en modo lectura, cuxo número de *hard links* é 1. O usuario efectivo do proceso coincide co propietario do ficheiro e tamén ten os permisos de acceso aos directorios *raíz, home* e *usr1*. Os cachés de datos e inodos están inicialmente baleiros e a entrada *usr1* está no sétimo bloque do directorio *home*, mentres que as outras entradas están no primeiro bloque dos seus directorios pai. Indique o seguinte sobre o anaco de código:

A process opens the previous file (in P1) *"/home/usr1/datos" (size* 8 Gbytes + 32 Mbytes + 8 Kbytes*)* in read mode, whose number of *hard links* is 1. The effective user of the process coincides with the owner of the file, and also has the access permissions to the directories *root*, *home* and *usr1*. The data and inode caches are initially empty and the entry *usr1* is in the seventh block of the directory *home*, while the other entries are in the first block of their parent directories. Indicate the following regarding the piece of code:

```
struct stat buf; char c1, c2;
chmod("/home/usr1/datos", 0642);
link ("/home/usr1/datos", "/home/usr1/datos2");
symlink("home/usr1/datos", "/home/user1/sim_link"); /*crea link simbólico sim_slink */
                                                    /* crea link simbólico sim_slink */
                                                    /* symbolic link sim_slink is created*/
int fd=open("/home/usr1/datos2", O_RDONLY); /* es la primera apertura de fichero del proceso */
                                            /* é a primeira apertura de ficheiro do proceso */
                                            /* it is the first file open of the process*/
int fd_sim=open("/home/usr1/sim_link", O_RDONLY);

lseek(fd, 16.000, SEEK_SET);   /*SEEK_SET indica que el desplazamiento se considera a partir del origen del fichero */
                               /* SEEK_SET indica que o desprazamiento considérase desde a orixe do ficheiro */
                               /* SEEK_SET specifies that the offset is considered from the origin of the file */
c1=fgetc(fd);
close(fd); close(fd_sim);
unlink("/home/usr1/datos"); unlink("/home/usr1/sim_link");
```

Cada apartado puntúa 0.1p/Each question scores 0.1p.

A. ¿Cuál es el valor asignado al descriptor de fichero *fd_sim*?:
Cal é o valor asignado ao descriptor de ficheiro *fd_sim*?:
What is the value assigned to the file descriptor *fd_sim*?: **4**

B. ¿Cuál es el número de accesos necesarios a disco, únicamente en el área de datos, en la apertura del fichero *"datos2"*?:
Cal é o número de accesos ao disco necesarios, só na área de datos, ao abrir o ficheiro *"datos2"*?:
What is the number of disk accesses required, only in the data area, when opening file *"datos2"*?: **9**

C. Indica el número de bloques que el SO necesita leer en disco para obtener el valor de *c1* en *fgetc(fd)*:
Indica o número de bloques que o SO debe ler desde o disco para obter o valor de *c1* en *fgetc (fd)*:
Indicate the number of blocks that the OS must read from disk to get the value of *c1* in *fgetc (fd)*: **1**

D. Después de ejecutar *chmod()*, indica los permisos del fichero que se imprimen en formato *rwxrwxrwx*:
Despois de executar *chmod(),* indica os permisos de ficheiro que se imprimen en formato *rwxrwxrwx*:
After executing *chmod(),* indicate the file permissions that are printed in *rwxrwxrwx* format: **rw- r-- -w-**

E. ¿Cuál es el número de hard links de *"datos2"* después de ejecutar los dos *unlink*?:
Cal é o número de ligazóns duros (*hard links*) de *"datos2"* despois de executar os dous *unlink*?:
What is the number of hard links of *"datos2"* after running the two *unlink*?: **1**

P3) En la partición de disco correspondiente, calcular qué número de bloque lógico corresponde al inodo del directorio *raíz* (se comienza a contar en el bloque lógico 0), y cuál bloque lógico al inodo del fichero *"datos"* (en P1), suponiendo lo siguiente:

i) El nº de inodo del "/" es el 2, y al fichero *"datos"* le corresponde el inodo nº 400 (los inodos se comienzan a numerar a partir de 1).
ii) El tamaño de un inodo es de 64 bytes (tamaño bloque = 8Kbytes).
iii) El *boot* ocupa 2 bloques y el *superbloque* 9 bloques.

Nº bloque lógico correspondiente al inodo de "/": **11** (0.1p)

Nº bloque lógico correspondiente al inodo de *"datos"*: **14** (0.2p)

Na partición de disco correspondente, calcular que número de bloque lóxico corresponde ao inodo do directorio *raíz* (o reconto comeza no bloque lóxico 0) e que bloque lóxico corresponde ao inodo do ficheiro *"datos"* (en P1), asumindo o seguinte:

i) O nº de inodo do "/" é 2, e ao ficheiro *"datos"* correspóndelle o inodo número 400 (os inodos comezan a numerarse desde 1).
ii) O tamaño dun inodo é de 64 bytes (tamaño do bloque = 8Kbytes).
iii) O *boot* ocupa 2 bloques e o *superbloque* 9 bloques.

Nº bloque lóxico correspondente ao inodo de "/": (0.1p)

Nº bloque lóxico correspondente ao inodo de *"datos"*: (0.2p)

In the corresponding file disk partition, calculate which logical block number corresponds to the *root* inode directory (counting starts at logical block 0), and which logical block corresponds to the inode of the file *"datos"* (in P1), assuming the following:

i) The inode number of the "/" is 2, and the inode number 400 (inodes start numbering from 1) corresponds to the file *"datos"*.
ii) The size of an inode is 64 bytes (block size = 8Kbytes).
iii) The *boot* occupies 2 blocks and the *superblock* 9 blocks.

Logical block number corresponding to the inode of "/": (0.1p)

Logical block number corresponding to the inode of *"datos"*: (0.2p)

P4) Indicar si es cierto/falso en cada pregunta.

Cada apartado puntúa 0.1p. Cada respuesta errónea puntúa -0.1. Cuestiones no respondidas no puntúan. La puntuación mínima de P4 es 0, es decir, en ningún caso P4 lleva a puntuación negativa para el total del examen.

A. El S.O. mantiene una copia en memoria principal del inodo de un fichero abierto, al menos hasta el último cierre del fichero. **C**

B. Al lincar un código con una librería estática, el código en memoria de las funciones necesarias de la librería se puede compartir entre varios procesos. **F**

C. Al ejecutar la función *perror()* de C se incrementa el tiempo de ejecución en modo usuario pero no en modo sistema. **F**

D. Es posible crear *soft links* entre diferentes sistemas de ficheros montados. **C**

E. La idea fundamental de un sistema de ficheros Unix basado en registro (*journaling file system*) es llevar control/registro de las asignaciones de los inodos a lo largo del tiempo. **F**

F. El *Buffer Cache* reduce el número de lecturas físicas sobre los discos montados, pero no el número de escrituras físicas. **F**

Indique se é verdadeiro/falso en cada pregunta.

Cada apartado puntúa 0.1p. Cada resposta incorrecta puntúa -0.1p. Cuestións non respondidas non puntúan. A puntuación mínima para P4 é 0, é dicir, en ningún caso P4 ten puntuación negativa para o exame total.

A. O S.O. mantén unha copia en memoria principal do inodo dun ficheiro aberto, polo menos ata o último peche do ficheiro. ____

B. Ao lincar un código cunha librería estática, o código en memoria das funcións necesarias da librería pódese compartir entre varios procesos. ____

C. Ao executar a función *perror()* de C increméntase o tempo de execución en modo usuario, pero non en modo sistema. ____

D. E posible crear *soft links* entre diferentes sistemas de ficheiros montados. ____

E. A idea fundamental dun sistema de ficheiros Unix basado en rexistro (*journaling file system*) é levar un control/rexistro das asignacións dos inodos ao longo do tempo. ____

F. A *Buffer Cache* reduce o número de lecturas físicas sobre os discos montados, pero non o número de escrituras físicas. ____

Indicate whether it is true/false for each question.

Each question scores 0.1p. Each wrong answer scores -0.1p. Unanswered questions do not score.The minimum score for P4 is 0, that is, in no case does P4 have a negative score for the total exam.

A. The O.S. keeps a copy of the inode of an open file in main memory, at least until the final closing of the file. ____

B. When linking code to a static library, the in-memory code of the necessary library functions can be shared between processes. ____

C. Executing the C *perror()* function increases the runtime in user mode, but not in system mode. ____

D. It is possible to create *soft links* between different mounted filesystems. ____

E. The fundamental idea of a Unix *journaling file system* is to keep track/record of inode assignments over time. ____

F. The *Buffer Cache* reduces the number of physical reads on mounted disks, but not the number of physical writes. ____

Apellidos:_____Nombre:_____ DNI: _____

Examen sin identificación completa NON se califica
Sistemas Operativos - GEI UDC. Xullo 2022. Memoria (2 puntos)
Esta parte NON admite entrega follas adicionais. Hay que contestar nas páxinas e espacios proporcionados.

1. Cada una de las preguntas siguientes son de responder V/F (Verdadero/Falso) o no responder. Respuesta correcta: +0.1; Respuesta incorrecta: -0.1. (0.1 x 10 = 1 punto) Puntuación mínima para esta pregunta: 0.
Cada unha das preguntas seguintes son de responder V/F (Verdadeiro/Falso) ou non responder. Resposta correcta: +0.1; Resposta incorrecta: -0.1. (0.1 x 10 = 1 punto) Puntuación mínima para esta pregunta: 0.
Answer T/F (True/False) or leave the answer blank. Correct answer: 0.1; Wrong answer: -0.1 (0.1 x 10 = 1 point) Minimum score for this question: 0.

1	2	3	4	5	6	7	8	9	10
F	F	F	F	F	F	V	V	F	F

1. **Una ventaja de un sistema de paginación multinivel con respecto a uno de un nivel es que se reducen las referencias de los procesos a páginas inválidas.**
Unha vantaxe dun sistema de paxinación de varios niveis fronte a unha paxinación dun só nivel é que redúcense as referencias do proceso a páxinas non válidas.
An advantage of a multi-level pagination system over a one-level one is that it has fewer invalid page references.

2. La TLB en un sistema con paginación y memoria virtual permite reducir las referencias a páginas inválidas de los procesos.
O TLB nun sistema con paxinación e memoria virtual permite reducir as referencias a páxinas non válidas dos procesos.
The TLB in a system with paging and virtual memory allows to reduce the references to invalid pages of the processes.

3. En un sistema con Tabla de Páginas Invertida hay una Tabla de Páginas Invertida para cada proceso.
Nun sistema con Táboa de Páxinas Invertida hai unha Táboa de Páxinas Invertida para cada proceso.
In an system with Inverted Page Table there is an Inverted Page Table for each process.

4. En un SO Linux la ventaja de implementar el espacio de intercambio como un archivo en vez de una partición dedicada, es que es más rápido el intercambio por usar el software del sistema de archivos.
Nun sistema operativo Linux, a vantaxe de implementar o espazo de intercambio como un ficheiro en lugar dunha partición dedicada é que e másі rápido o intercambio por usar o software do sistema de ficheiros,.
In a Linux OS the advantage of implementing the swap space as a file instead of a dedicated partition is that makes the swap faster because it uses the file system software

5. Al volver con éxito de una llamada al sistema fork() en un SO Linux, con mecanismo copy-on-write, se han duplicado las tablas de páginas y las páginas de lectura/escritura del proceso padre en el proceso hijo.

Ao regresar con éxito dunha chamada de sistema fork () nun sistema operativo Linux con mecanismo de copia en escritura, as táboas de páxinas e as páxinas de lectura/escritura do proceso pai duplicáronse no proceso fillo.
On successful return from a fork () system call on a Linux OS with copy-on-write mechanism, the page tables and read/write pages of the parent process have been duplicated in the child process.

6. Una variable estática no puede almacenar el valor devuelto por una llamada exitosa a la función de librería malloc() para obtener memoria del heap.
Unha variable estática non pode almacenar o valor devolto por unha chamada exitosa á función da biblioteca malloc () para obter memoria do heap.
A static variable cannot store the value returned by a successful call to the malloc() library function to get memory from the heap.

7. Considere un sistema con memoria virtual, paginación por demanda, asignación fija de N marcos de memoria a cada proceso y algoritmo de reemplazo FIFO segunda oportunidad. Para un proceso con N-1 páginas virtuales se producen N-1 o menos fallos de página.
Considere un sistema con memoria virtual, paxinación baixo demanda, asignación fixa de N marcos de memoria a cada proceso e o algoritmo de reemplazo FIFO segunda oportunidade. Para un proceso con N-1 páxinas virtuais, ocorren N-1 ou menos fallos de páxina.
Consider a system with virtual memory, on-demand paging, fixed allocation of N memory frames to each process, and the second chance FIFO replacement algorithm. For a process with N-1 virtual pages, there are N-1 or fewer page faults.

8. Considere un hardware básico para un sistema de memoria que proporciona dos registros límite (superior e inferior). No proporciona registro base, ni ningún otro mecanismo de segmentación o paginación. En este sistema puede ejecutarse código absoluto y código con relocalización estática de direciones.
Considere un hardware básico para un sistema de memoria que proporciona dous rexistros límite (superior e inferior). Non proporciona rexistro base, nin ningún outro mecanismo de segmentación ou paginación. Neste sistema pódese executar código absolutos e códigos con relocalización estática de direccións.
Consider a basic hardware for a memory system that provides two limit registers (upper and lower). It does not provide a base register, or any other segmentation or paging mechanism. Absolute code and code with static address relocation can be executed on this system.

9. Una función que entre sus variables locales tiene alguna variable estática, no usa el stack de usuario.
Unha función que entre as súas variables locais ten algunha estática, non usa o stack de usuario.
A function that has some static variables among its local variables, does not use the user stack.

10. Considere direcciones virtuales de 48 bits y tabla de páginas en 1 nivel con entradas de 4 bytes y páginas de 4 Kbytes. La tabla de páginas ocupa 2^{24} páginas.
Considere direccións virtuais de 48 bits e táboa de páxinas dun nivel con entradas de 4 bytes e páxinas de 4 kbytes. A taboa de páxinas ocupa 2^{24} páxinas.
Consider 48 bits virtual addresses and a 1 level Page Table with 4 byte entries and 4 Kbyte pages. The page table occupies 2^{24} pages.

2. (0.4 puntos) Imagine un sistema con 64 Kbytes de memoria física. Supongamos que el sistema operativo y sus estructuras de datos ocupan 16 Kbytes y hay dos procesos, uno con 19 Kbytes de código y 5 Kbytes de datos y el otro con 15 Kbytes de código y 8 Kbytes de datos. A) ¿Pueden estos procesos caber en la memoria física si se utilizan páginas de 2 Kbytes? B) ¿Qué pasa si se usa segmentación pura?
Imaxina un sistema con 64 Kbytes de memoria física. Supoñamos que o SO e as súas estruturas de datos ocupan 16 Kbytes e hai dous procesos, un con 19 Kbytes de código e 5 Kbyes de datos e o outro con 15 Kbytes de código e 8 Kbytes de datos. A) Poden caber estes procesos na memoria física se se usan páxinas de 2Kbytes? B) E se se usa a segmentación pura?
Imagine a system with 64Kbytes of physical memory. Suppose the OS and its data structures occupy 16Kbytes and there are two processes, one with 19Kbytes of code and 5Kbyes of data and the other with 15Kbytes of code and 8Kbytes of data. A) Can these processes fit in physical memory if 2Kbytes pages are used? B) What if pure segmentation is used?

A) Paginación: no. La memoria física se asignará en marcos de página completos. Por lo tanto, el primer proceso necesita 10 + 3 marcos, el segundo proceso necesita 8 + 4 y el sistema operativo necesita 8, para un total de 33. Solo hay 32 marcos disponibles.
B) Segmentación: sí. Podemos asignar el sistema operativo y cada código de proceso y datos a un segmento separado. Podemos asignar los segmentos para que sean exactamente tan grandes como deben ser. El requisito de memoria total es 16 Kbytes + 19 Kbytes + 5 Kbytes + 15 Kbytes + 8 Kbytes = 63 Kbtes <64 Kbytes.

3. (0.35 ptos) Considere una arquitectura de memoria con un TLB con tiempo de acceso de 1ns (tanto para un fallo como un acierto en el TLB), tablas de páginas en 4 niveles con 512 entradas cada una. El tiempo de acceso a memoria es 100ns. Suponga que no hay fallos de página a disco. ¿Qué porcentaje de acierto en el TLB se necesita para tener un tiempo efectivo de acceso a memoria de 110 ns?: _____
Considere unha arquitectura de memoria cun tempo de acceso TLB de 1ns (tanto para un TLB acertado como para fallar), táboas de páxinas de 4 niveis con 512 entradas cada unha. O tempo de acceso á memoria é de 100 ns. Supoña que non hai fallos de páxina a disco. Que porcentaxe de éxito no TLB se necesita para ter un tempo de acceso efectivo á memoria de 110 ns?:
_____
Consider a memory architecture with a 1ns access time TLB (for both a TLB hit and miss), 4-level page tables with 512 entries each. Memory access time is 100ns. Assume there are no page-to-disk faults. What percentage of success in the TLB is needed to have an effective memory access time of 110 ns ?: _____

Sea HR (hit rate) la tasa de acierto en el TLB
HR x 101 + (1- HR) x 501 = 110
391 = 400 HR
HR = 0,9775 (97,75%)

3. (0.25 ptos) Cada una de las preguntas siguientes son de responder V/F (Verdadero/Falso) o no responder. Respuesta correcta: +0.05; Respuesta incorrecta: -0.05. (0.05 x 5 = 0.25 punto) Puntuación mínima para esta pregunta: 0.
Cada unha das preguntas seguintes son de responder V/F (Verdadeiro/Falso) ou non responder. Resposta correcta: +0.05 Resposta incorrecta: -0.05. (0.05 x 5 = 0.25 puntos) Puntuación mínima para esta pregunta: 0.
Answer T/F (True/False) or leave the answer blank. Correct answer: 0.05; Wrong answer: -0.05 (0.05 x 5 = 0.25 point) Minimum score for this question: 0.

1	2	3	4	5
F	F	V	V	V

Considere el esquema de gestión de memoria con un registro base y un registro límite.
1. No proporciona relocalización dinámica
2. No proporciona protección de la memoria
3. El espacio lógico de direcciones de un proceso es contiguo
4. Puede tener el problema de fragmentación externa
5. Un proceso sólo puede tener un segmento

Considere o esquema de xestión de memoria con un rexistro base e un rexistro límite.
1. Non proporciona relocalización dinámica
2. Non proporciona protección da memoria
3. O espazo de direccións lóxicas dun proceso é contiguo
4. Pode ter o problema da fragmentación externa
5. Un proceso só pode ter un segmento

Consider the memory management scheme with one base and one imit registers.
1. Does not provide dynamic relocation
2. Does not provide memory protection
3. The logical address space of a process is contiguous
4. It can have the problem of external fragmentation
5. A process can only have one segment

APELLIDOS:_____ **NOMBRE:**_____ **DNI:** _____

USE LETRAS MAYÚSCULAS EN APELLIDOS Y NOMBRE

Sistemas Operativos - Grado Ingeniera Informática UDC. Junio de 2022

Parte E/S (1.5 puntos).

1: (0.5 puntos/points)

Respóndase V/F (Verdadero/Falso) o deje en blanco. Respuesta correcta: +0.1; Respuesta incorrecta: -0.1. Puntuación mínima en esta pregunta: 0.

Responda V/F (Verdadeiro/Falso) ou deixe en branco. Resposta correcta: +0.1; Resposta incorrecta: -0.1. Puntuación mínima nesta pregunta: 0.

Answer T/F (True/False) or leave as blank. Correct answer: +0.1; Wrong answer: -0.1. Minimum score for this question: 0.

A	B	C	D	E
V	F	F	F	F

← Responde V/F aquí, o deja en blanco

Responde V/F aquí, ou deixa en branco

either answer T/F here, or leave blank

A- La capa del subsistema de E/S denominada "software independente del dispositivo" se encarga de calcular qué bloque de disco contiene el i-ésimo byte de un fichero.

A capa do subsistema de E/S denominada "software independente do dispositivo" está a cargo de calcular cal é o bloque de disco que contén o i-ésimo byte dun ficheiro.

The layer of the I/O subsystem named "Device Independent Software" computes which is the disk block containing the i-th byte of a given file.

B- En una operación de salida de datos, el registro de control de un controlador de dispositivo permite saber si el dato a enviar ya ha sido transferido al dispositivo.

Nunha operación de saída de datos, o rexistro de control dun controlador de dispositivo permite saber se o dato a enviar xa foi transferido ao dispositivo.

During a data-output transference, the control register of a device controller permits us to know if the piece of data to be sent/written has already been transferred to the device

C- Un disco duro con 4 caras y 4096 pistas por cara, tiene 1024 cilindros.

Un disco duro con 4 caras e 4096 pistas por cara, ten 1024 cilindros.

A hard disk with 4 sides and 4096 tracks in each side contains 1024 cylinders.

D- Los *minor numbers* de los dispositivos /dev/sda1 y /dev/sdb1 son idénticos.

Os minor numbers dos dispositivos /dev/sda1 e /dev/sdb1 son idénticos.

Devices /dev/sda1 and /dev/sdb1 have the same minor numbers.

E- El uso de dispositivos "mapeados a memoria" conlleva que la CPU deba utilizar instrucciones especiales (e.j. IN en intel) para la comunicación con el dispositivo.

O uso de dispositivos "mapeados a memoria" conleva que a CPU deba utilizar instrucciónss especiais (e.x. IN en intel) para a comunicación co dispositivo.

Using "memory-mapped devices" implies that the CPU must use special instructions (e.g. IN in intel) to communicate with the device.

2: (0.5 puntos/points)

La cola de peticiones de E/S para acceder a cilindros de un disco contiene las peticiones [166, 100, 160, 77, 135, 12], tras haber atendido las peticiones [110] y [115] (en este orden). Indique en qué orden se atenderán las restantes peticiones.

A cola de peticións de E/S para acceder a cilindros dun disco contén as peticións: [166, 100, 160, 77, 135, 12] tras ter atendido as peticións [110] e [115] (nesta orde). Indíquese en que orde serán atendidas as restantes peticións.

The I/O requests queue for accessing cylinders within a disk contains the requests [166, 100,160, 77, 135, 12]. Requests [110] and [115] have already been attended (in this order). Indicate in which order the pending requests will be attended.

Algoritmo / *algorithm*	Cilindros accedidos / *cylinders accessed*							
SSTF	110	115	100	77	135	160	166	12
C-LOOK	110	115	135	160	166	12	77	100
SCAN	110	115	135	160	166	100	77	12

3: (0.5 puntos/points)

El fichero "a.dat" contiene "ABCDEFGHIJKL\n". ¿Qué se escribe en pantalla en las llamadas a printf de las líneas lin.08 y lin17?

O ficheiro "a.dat"contén "ABCDEFGHIJKL\n". Que se escribe en pantalla nas chamadas a printf das liñas lin.08 e lin17?

File "a.dat" contains "ABCDEFGHIJKL\n". What is written into the screen in the calls to printf within lines lin.08 and lin.17?

```
//lin.01:   int  main(int argc, char *argv[])
//lin.02:   {errno=0;
//lin.03:    struct aiocb aio;     long ret1, ret2, ret3;
//lin.04:    char BUF1[5]={'\0','\0','\0','\0','\0'};
//lin.05:    char BUF2[5]={'\0','\0','\0','\0','\0'};
//lin.06:    int fd = open("a.dat", O_RDONLY);
//lin.07:    ret1 = pread(fd, BUF1, 4,2);
//lin.08:    printf("%s--%ld",BUF1,ret1);          /**/
//lin.09:
//lin.10:    aio.aio_fildes  = fd;
//lin.11:    aio.aio_buf     = BUF2; aio.aio_nbytes  = 4;
//lin.12:    aio.aio_reqprio = 0;    aio.aio_offset  = 1;
//lin.13:    aio.aio_sigevent.sigev_notify = SIGEV_NONE;
//lin.14:    ret2= aio_read(&aio);
//lin.15:    while(aio_error(&aio)==EINPROGRESS);
//lin.16:    ret3 = aio_return(&aio);
//lin.17:    printf("%s--%ld",BUF2,ret2);         /**/
//lin.18:   }
```

Nota: asúmase que no hubo ningún error de ejecución.

Nota: Asúmase que non houbo erros durante a execución.

Note: Let us assume no errors were reported during the execution.

printf lin.08: =

C	D	E	F	-	-	4

← Contesta aquí / *fill your answer here*

Pon un char en cada celda / *write a char in each cell*

Breve justificación / *breve xustificación / brief explanation* ➔

pread lee 4 chars a partir de la posición 2 y los almacena en BUF1 → CDEF
ret1 guarda el valor devuelto por pread; esto es, cuántos bytes se han leído → 4

printf lin.17: =

B	C	D	E	-	-	0

← Contesta aquí / *fill your answer here*

Pon un char en cada celda / *write a char in each cell*

Breve justificación / *breve xustificación / brief explanation* ➔

BUF2 contiene 4 chars del fichero, leídos a partir del offset 1 del fichero → BCDE
ret2 guarda un 0 (indica si se han producido errores en aio_read: no → 0 (-1 si hay error al encolar petición)
ret3 guarda cuántos bytes se han leído = 4 (pero no se muestra en el printf).

APELLIDOS:_____ NOMBRE:_____ DNI: _____

Sistemas Operativos (PROCESOS) – Grado Ingeniería Informática UDC. Julio 2022.
-Apellidos en MAYUSCULA.
-Examen sin nombre equivale a NO PRESENTADO).

Q1.-(0.8 puntos) Responda Verdadero/Falso (o no conteste) a cada pregunta. Respuesta correcta 0.1 Respuesta incorrecta -0.1. La puntuación mínima para esta pregunta es 0. (rellenar en el espacio correspondiente de la siguiente tabla)/*Answer V/F (True/False) or leave the answer blank. Correct answer: 0.1;Wrong answer: -0.1 Minimum score for this question: 0/* Cada unha das preguntas seguintes son de responder V/F (Verdadeiro/Falso) ou non responder. Resposta correcta: +0.1; Resposta incorrecta: -0.1. Puntuación mínima para esta pregunta: 0

1	2	3	4	5	6	7	8
V	***F***	***V***	***V***	***F***	***V***	***F***	***F***

1-En un proceso DEL ROOT, las llamadas a funciones de la librería matemática son en modo usuario
A call to a math library function in a process of user ROOT, is done in user mode
Nun proceso que DO ROOT, as chamadas a funciones da libreria matemática fanse en modo usuario

3-Mientras quede memoria (e intercambio) libre en el sistema, una llamada a fork() crea un proceso
As long as there's free memory (and swap) on the system, calling fork() creates a process
Mentres haxa memoria libre (e intercambio) no sistema, unha chamada a fork() crea un proceso

3-En un sistema tipo UNIX donde hay 3 usuarios ejecutando el navegador: el root y dos usuarios normales, hay una sola copia del código del navegador en memoria
In a UNIX type system where there are 3 different users (the root and two normal users) executing the web browser, there is only one copy of the web browsers code in memory
Nun sistema tipo UNIX onde hai 3 usuarios diferentes (un deles é o root) que executan o navegador web, hai unha soa copia na memoria do código do navegador web

4.-La llamada fork() nunca devuelve 1
The fork() system call does never return 1
A chamada ao sistema fork() nunva devolve o valor 1

5.-El root puede crear link simbólicos entre procesos de distintos usuarios
The root user can create symboling links between processes from diferent users
O root pode crear enlaces simbólicos entre procesos de diferentes usuarios

6-El código de la rutina de servicio de la interrupción de teclado se ejecuta SIEMPRE en modo kernel
The keyboard interrupt service routine code is ALWAYS executed in kernel mode
O código de rutina do servizo de interrupción do teclado execútase SEMPRE en modo kernel

APELLIDOS:_____ **NOMBRE:**_____ **DNI:** _____

7-Si "file" existe y tiene permiso de ejecución la llamada *execv* (char * file, char * args[]) no devuelve nada
*As long as "file" exists and has execute permission, the execv (char * file, char * args[]) will return nothing*
Si "file" existe e ten permisos de execución, a chamada execv (char * file, char *args[]) non devolve nada

8-Un fichero setuid del root solo puede ser ejecutado por un proceso con credencial efectiva del root
A root setuid file can only be executed by a process with an effective root credential.
Un ficheiro setuid do root só pode ser executado por un proceso coa credencial efectiva do root.

Q2.-(0.6 puntos) Sea el siguiente código en C, con todos los includes necesarios y que compila correctamente. Se supone que ni execl ni fork producen error
Consider the following C code, with all the necessary includes and that compiles correctly. It is assumed that neither execl nor fork produce any error
Sexa o seguinte código en C, con todas os includes necesarios e que compila correctamente. Supoñemos que nin execl nin fork producen erro nungún

```
#define VECES 3

int Valor;
main(int argc, char * argv[])
{
  pid_t pid;
  Valor=2022;
  for (i=0; i<VECES; i++){
      pid=fork()+fork();
      if (execl("./b.out", "./b.out", NULL)==-1)
          exit(0);
      printf ("/%d/\n", Valor);
  }
}
```

```
/*Código de b.out*/

int Valor;
main (int argc, char *argv[])
{
    printf ("/%d/\n",Valor);
}
```

a) ¿cuántas líneas de salida produce? *How many output lines does it produce?.* ¿cantas liñas de saida produce?
4
b)¿cuántas de ellas son «/0/*»?-*How many of them are «/0/*»*? ¿cántas delas son «/0/*»?
4.
En la primera iteración se crean 3 procesos (con lo que hay 4). Los 4 llegan al exec y reemplazan su código cpn b.out, con lo que imprimen /0/ (variable externa sin inicializar)

APELLIDOS:_____ NOMBRE:_____ DNI: _____

Q3.-(0.6 puntos correcta, -0.2 puntos incorrecta) Un sistema tiene tres tipos de procesos: *A system supports three kinds of processes:* Un sistema ten tres tipos de procesos

*Procesos en tiempo real (RT), máximas prioridades en el sistema, entre ellos se planifican por prioridades estáticas apropiativas (mayor número mayor prioridad). *Real Time Processes, (higher system priorities) with preemptive priority scheduling (the higher the number, the higher the priority)* Procesos en tempo real (RT), (prioridades mais altas do sistema), entre eles prioridades estáticas apropiativas (número maior prioridade maior),

*Procesos interactivos (IT), que corresponden a una prioridad estática de 0 y entre ellos se planifican por round-robin de cuanto 300. *Interactive processes (correspond to a static priority of 0). they are scheduled in a round-robin fashion with a 300ms quantum.* Procesos interactivos (IT), que corresponden a unha prioridade estática de 0 e entre eles planifícanse con round-robin de quantum 300,

*Procesos "idle" (ID), la mínima prioridad en el sistema, solo se ejecutan si no hay listo en el sistema ninguno de los otros procesos y entre ellos se planifican por SRTF. *"Idle" processes, only get executed when theres no other ready to run process in the system. They use a SRTF scheduling.* Procesos "idle" (ID), a prioridade máis baixa do sistema, só se executan se ningún dos outros procesos está listo no sistema. Entre eles planifícanse por SRTF.

En un sistema hay los siguientes procesos (AT representa el instante de llegada). Las ráfagas se representan CPU-(E/S)-CPU-(E/S)....-CPU. *A system has the following processes (AT means arrival time). Burts are represented CPU-(I/O)-CPU-(I/O).....-CPU*. Nun sistema hai os seguintes procesos (AT é o instante de chegada). As ráfagas represéntanse CPU-(I/O)-CPU-(I/O)....-CPU.

Proceso	TIPO	PRI	AT	Ráfagas/bursts (ms)
A	RT	20	0	300-(700)-200
B	RT	25	200	300-(900)-100
C	RT	30	400	100-(300)-100-(400)-100
D	IT	0	900	400-(200)-500
E	IDLE	-1	500	400
F	IDLE	-1	1200	100-(100)-100

Vemos cuatro posibles planificaciones. ¿Cual es la correcta?.

Which of the following scheduling scenarios is correct?

¿Cal das seguintes é a planificación correcta?

La planificación correcta es la 3. En la 1 y la 4 las prioridades de los procesos en tiempo real son no apropiativas. En la 2, en el instante 1200, F, proceso idle, no debería ejecutarse hasta que D acabe su ráfaga, no su cuanto (procesos idle solo se ejecutan si no hay nadie listo, D acaba su quanto, y está listo)

APELLIDOS:_____ NOMBRE:_____ DNI: _____

Planificación/Scheduling 1:

CPU	A	A	A	B	B	B	C	E	E	D	C	A	A	D	D	C	B	D	F	E	D	D	D	D	D	F	E
E/S				A	A	A	AB	ABC	ABC	ABC	B	BC	BC	BC	BC				D	DF								

Planificación/ Sheduling 2:

CPU	A	A	B	B	C	B	A	E	C	D	D	D	F	C	A	B	A	D	F	E	D	D	D	E	E	D	D
E/S						C	CB	CBA	BA	CBA	CBA	CBA	CBA	FBA	B				D	D								

Planificación/Scheduling 3:

CPU	A	A	B	B	C	B	A	E	C	D	D	D	D	C	A	B	A	D	D	D	D	D	F	E	F	E	E
E/S						C	CB	CBA	BA	CBA	CBA	CBA	CBA	DBA	DB									F			

Planificación/Scheduling 4:

CPU	A	A	A	B	B	B	C	E	E	D	C	A	A	D	D	C	B	D	F	E	D	D	D	F	D	D	E
E/S				A	A	A	AB	ABC	ABC	ABC	B	BC	BC	BC	BC				D	DF							